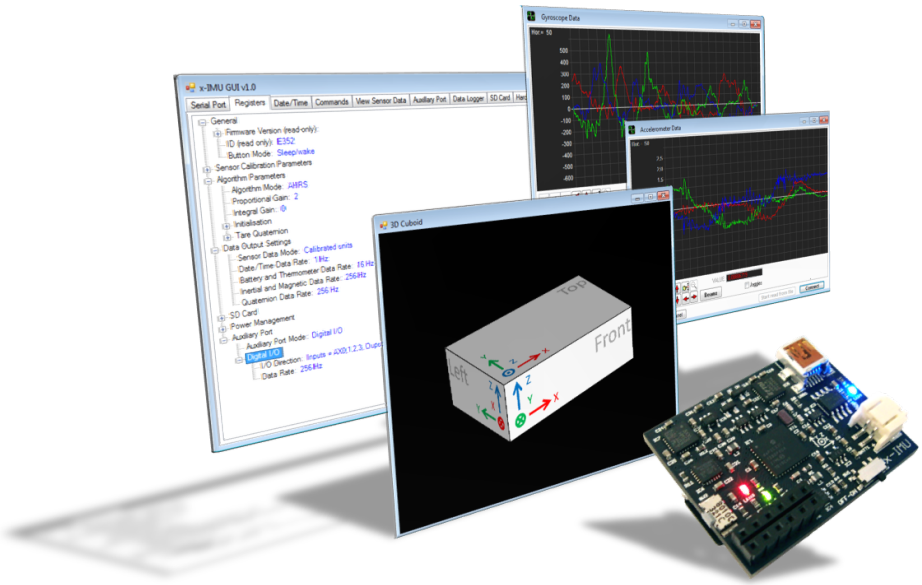
# x-IMU User Manual 2.1

x-io Technologies

May 31, 2011



| Version | Date | Notes |
|---|---|---|
| 2.1 | 17/05/2011 | - Minor corrections spelling/wording/layout/appearance througout.<br>- IMU/AHRS algorithm section has been updated.<br>- TODO: link to API docuemtnation etc. |
| 2.0 | 17/05/2011 | - Minor corrections spelling/wording/layout/appearance througout.<br>- Documented changes introduced by firmware 5.0, API 8.0 and GUI 7.0. See individual release notes for details.<br>Compatible versions: firmware 5.x, GUI 7.x, API 8.x. |
| 1.0 | 10/05/2011 | - Initial release.<br>Compatible versions: firmware 4.x, GUI 6.x, API 7.x. |

Table 1: Document version history

## Disclaimer

*The x-IMU and associated software are provided in an 'as in' condition. No warranties, whether express, implied or statutory, including but not limited to implied warranties of merchantability and fitness for a particular purpose apply. x-io Technologies shall not in any circumstances, be liable for special, incidental or consequential damages, for any reason whatsoever.*

# Contents


**1 x-IMU overview** **4**
- 1.1 x-IMU Features . . . 4
- 1.2 x-IMU Software . . . 5

**2 Getting started** **6**

**3 Hardware overview** **7**
- 3.1 Power switch . . . 7
- 3.2 Push button . . . 7
- 3.3 LEDs . . . 8
  - 3.3.1 Status LED (Green) . . . 8
  - 3.3.2 SD card LED (Amber) . . . 8
  - 3.3.3 Bluetooth LED (Blue) . . . 8
  - 3.3.4 Charging LED (Red) . . . 8
- 3.4 USB socket . . . 8
- 3.5 Micro-SD card socket . . . 8
- 3.6 Bluetooth radio . . . 8
- 3.7 Battery connector . . . 8

**4 Software overview** **9**
- 4.1 x-IMU GUI . . . 9
  - 4.1.1 Tab page: Serial port . . . 9
  - 4.1.2 Tab page: Registers . . . 9
  - 4.1.3 Tab page: Date/time . . . 10
  - 4.1.4 Tab page: Commands . . . 11
  - 4.1.5 Tab page: View sensor data . . . 11
  - 4.1.6 Tab page: Auxillary port . . . 13
  - 4.1.7 Tab page: Data logger . . . 14
  - 4.1.8 Tab page: SD card . . . 15
  - 4.1.9 Tab page: Hard-iron calibration . . . 16
  - 4.1.10 Tab page: Upload firmware . . . 16
- 4.2 x-IMU API . . . 17

**5 USB** **17**
- 5.1 Installing USB drivers . . . 17
- 5.2 USB bandwidth . . . 18

**6 Bluetooth** **18**
- 6.1 Pairing the x-IMU with a Bluetooth host . . . 19
- 6.2 Bluetooth LED . . . 20
- 6.3 Bluetooth bandwidth . . . 21
- 6.4 Optimising Bluetooth performance . . . 21

**7 SD card** **21**
- 7.1 Creating and closing files . . . 22
- 7.2 SD card LED . . . 22
- 7.3 SD card bandwidth . . . 22
- 7.4 Magnetic distortions from the SD card socket . . . 22

**8 Real-time clock and calendar** **22**
- 8.1 Maintaining clock power . . . 23

**9 Sensors** **23**

9.1 Battery voltmeter . . . 23
9.2 Thermometer . . . 23
9.3 Gyroscope . . . 24
9.4 Accelerometer . . . 24
9.5 Magnetometer . . . 25

**10 Sensor calibration** **25**

**11 IMU/AHRS algorithms** **25**
11.1 Principle of operation . . . 26
11.2 Algorithm parameters . . . 26
11.2.1 Algorithm mode . . . 26
11.2.2 Algorithm gains . . . 26
11.2.3 Algorithm magnetic feild rejection . . . 26
11.2.4 Algorithm tare quaternion . . . 26
11.3 Algorithm operations . . . 26
11.3.1 Algorithm initialise . . . 26
11.3.2 Algorithm tare . . . 26
11.3.3 Algorithm clear tare . . . 26
11.3.4 Algorithm initialise then tare . . . 26
11.4 Optimising algorithm performance . . . 26

**12 Power management** **26**
12.1 External supply . . . 26
12.2 Battery and charging . . . 27
12.3 Low battery voltage detection . . . 27
12.4 Sleep mode . . . 27
12.5 Sleep timer . . . 27
12.6 Motion triggered wake up . . . 28
12.7 Tips for minimising power consumption . . . 28

**13 Auxiliary port** **28**

**14 Communication protocol** **28**

**15 Commands** **28**
15.1 Individual commands . . . 29
15.1.1 Null command . . . 29
15.1.2 Reset . . . 29
15.1.3 Sleep . . . 29
15.1.4 Reset sleep timer . . . 29
15.1.5 Sample gyroscope bias at temperature 1 . . . 29
15.1.6 Sample gyroscope bias at temperature 2 . . . 29
15.1.7 Calculate gyroscope bias parameters . . . 30
15.1.8 Look up accelerometer bias and sensitivity . . . 30
15.1.9 Measure magnetometer bias and sensitivity . . . 30
15.1.10 Algorithm initialise . . . 30
15.1.11 Algorithm tare . . . 30
15.1.12 Algorithm clear tare . . . 30
15.1.13 Algorithm initialise then tare . . . 31

**16 Error messages** **31**
16.1 Individual error messages . . . 31
16.1.1 NoError . . . 31
16.1.2 Low battery . . . 31

16.1.3 USB receive buffer overrun . . . 31
16.1.4 USB transmit buffer overrun . . . 31
16.1.5 Bluetooth receive buffer overrun . . . 31
16.1.6 Bluetooth transmit buffer overrun . . . 32
16.1.7 SD card write buffer overrun . . . 32
16.1.8 Too few bytes in packet . . . 32
16.1.9 Too many bytes in packet . . . 32
16.1.10 Invalid checksum . . . 32
16.1.11 Unknown packet header . . . 32
16.1.12 Invalid number of bytes for packet header . . . 32
16.1.13 Invalid register address . . . 33
16.1.14 Register read-only . . . 33
16.1.15 Invalid register value . . . 33
16.1.16 Invalid command . . . 33
16.1.17 Gyroscope not stationary . . . 33
16.1.18 Magnetometer saturation . . . 33
16.1.19 Incorrect auxiliary port mode . . . 33

**17 Registers** **33**
17.1 Reading and writing registers . . . 34
17.2 Individual register details . . . 34
17.2.1 Firmware version major number . . . 34
17.2.2 Firmware version minor number . . . 34
17.2.3 Device ID . . . 34
17.2.4 Button mode . . . 34
17.2.5 Battery voltmeter sensitivity . . . 34
17.2.6 Battery voltmeter bias . . . 35
17.2.7 Thermometer sensitivity . . . 35
17.2.8 Thermometer bias . . . 35
17.2.9 Gyroscope x-axis sensitivity . . . 35
17.2.10 Gyroscope y-axis sensitivity . . . 35
17.2.11 Gyroscope z-axis sensitivity . . . 35
17.2.12 Gyroscope x-axis bias . . . 35
17.2.13 Gyroscope y-axis bias . . . 36
17.2.14 Gyroscope z-axis bias . . . 36
17.2.15 Gyroscope x-axis bias temperature sensitivity . . . 36
17.2.16 Gyroscope y-axis bias temperature sensitivity . . . 36
17.2.17 Gyroscope z-axis bias temperature sensitivity . . . 36
17.2.18 Gyroscope sample 1 - Temperature . . . 36
17.2.19 Gyroscope sample 1 - x-axis bias . . . 37
17.2.20 Gyroscope sample 1 - y-axis bias . . . 37
17.2.21 Gyroscope sample 1 - z-axis bias . . . 37
17.2.22 Gyroscope sample 2 - Temperature . . . 37
17.2.23 Gyroscope sample 2 - x-axis bias . . . 37
17.2.24 Gyroscope sample 2 - y-axis bias . . . 38
17.2.25 Gyroscope sample 2 - z-axis bias . . . 38
17.2.26 Accelerometer full-scale . . . 38
17.2.27 Accelerometer x-axis sensitivity . . . 38
17.2.28 Accelerometer y-axis sensitivity . . . 38
17.2.29 Accelerometer z-axis sensitivity . . . 38
17.2.30 Accelerometer x-axis bias . . . 39
17.2.31 Accelerometer y-axis bias . . . 39
17.2.32 Accelerometer z-axis bias . . . 39
17.2.33 Magnetometer full-scale . . . 39

17.2.34 Magnetometer x-axis sensitivity . . . 39
17.2.35 Magnetometer y-axis sensitivity . . . 40
17.2.36 Magnetometer z-axis sensitivity . . . 40
17.2.37 Magnetometer x-axis bias . . . 40
17.2.38 Magnetometer y-axis bias . . . 40
17.2.39 Magnetometer z-axis bias . . . 40
17.2.40 Magnetometer x-axis hard-iron bias . . . 40
17.2.41 Magnetometer y-axis hard-iron bias . . . 41
17.2.42 Magnetometer z-axis hard-iron bias . . . 41
17.2.43 Algorithm mode . . . 41
17.2.44 Algorithm proportional gain . . . 41
17.2.45 Algorithm integral gain . . . 41
17.2.46 Algorithm initial proportional gain . . . 41
17.2.47 Algorithm initilasation period . . . 42
17.2.48 Algorithm minimum valid magnetic field magnitude . . . 42
17.2.49 Algorithm maximum valid magnetic field magnitude . . . 42
17.2.50 Tare quaternion (element 0) . . . 42
17.2.51 Tare quaternion (element 1) . . . 42
17.2.52 Tare quaternion (element 2) . . . 42
17.2.53 Tare quaternion (element 3) . . . 43
17.2.54 Sensor data mode . . . 43
17.2.55 Date/time data output rate . . . 43
17.2.56 Battery and thermometer data output rate . . . 43
17.2.57 Inertial and magnetic data output rate . . . 43
17.2.58 Quaternion data output rate . . . 44
17.2.59 SD card new file name . . . 44
17.2.60 Battery shotdown voltage . . . 44
17.2.61 Sleep timer . . . 44
17.2.62 Motion trigger wake up . . . 44
17.2.63 Bluetooth power . . . 44
17.2.64 Auxiliary port mode . . . 45
17.2.65 Digital I/O direction . . . 45
17.2.66 Digital I/O data output rate . . . 45

# 1 x-IMU overview

The x-IMU was designed to be the most versatile IMU/AHRS product available. Its host of on-board sensors, algorithms and configurable 8-channel auxiliary port make the x-IMU both a powerful sensor and controller. Communication is enabled via USB or Bluetooth for wireless applications. The on-board SD card, battery charger (via USB), real-time clock/calendar and motion trigger wake up also make the x-IMU an ideal stand-alone data logger.

The open source x-IMU GUI allow users configure all internal x-IMU settings, view sensor data in real-time and export data to software such as MATLAB and Microsoft Excel. Custom user software may be developed using the x-IMU API.

## 1.1 x-IMU Features

**On-board sensors**

- Triple axis 16-bit gyroscope (2000 °/s range)
- Triple axis 12-bit accelerometer (configurable range up to 8 g)
- Triple axis 12-bit magnetometer (configurable range up to 8.1 G)
- 16-bit thermometer
- Battery level
- Calibrated and temperature compensated (gyroscope only)
- Configurable sample rates up to 256 Hz

**On-board algorithms**

- IMU algorithm for real-time measurement of pitch and roll relative to the Earth surface, immune to magnetic inferences
- AHRS algorithm for real-time measurement of full orientation relative to the Earth coordinate frame
- Sensor calibration algorithms for user maintenance

**Connectivity**

- USB
- Bluetooth (class 1, 100m range, SPP)
- Micro-SD card (supports SDHC and FAT16/32 for capacities up to 2 TBs)

**Power options**

- USB
- LiPo battery (on-board charging via USB)
- External source (3.6 V to 7.7 V)
- Low power consumption (35 mA to 100 mA dependant on settings and usage, 130 A sleep mode)

**Low profile**

- Dimensions: 33 × 42 × 10 mm
- Weight: 12g

**Other features**

- Motion triggered wake-up and sleep
- Real-time clock and calendar
- Configurable command button
- Configurable 8 channel auxiliary port

**Auxiliary port modes**

- Digital I/O (outputs controlled via USB or Bluetooth)

## 1.2 x-IMU Software

The x-IMU GUI (Graphical User Interface) provides interface to all features and functionality of the x-IMU via the x-IMU API. The x-IMU GUI is open source and is intended to serve as a comprehensive template for those using the x-IMU API to develop their own applications. Additional software examples using the x-IMU API for different applications can be found on the x-IMU Examples webpage.

**Features**

- View, edit and backup all internal x-IMU settings
- Real-time 2D and 3D data graphics
- Control panels for auxiliary port
- Data logger and file converter for exporting data; e.g. to MATLAB, Microsoft Excel, etc.
- Magnetic calibration tools
- Firmware bootloader to access new features in future x-IMU firmware versions

# 2 Getting started

1. Install the USB drivers (5.1) and/or pair the x-IMU as a Bluetooth device (6.1).
2. Download and install the latest version of the x-IMU GUI.
3. Connect to the x-IMU via the *Serial port* tab page (4.1.1) of the x-IMU GUI.

The x-IMU and accompanying software were designed to be versatile and intuitive. New users are advised to explore the x-IMU through the x-IMU GUI and consult this user manual when required. If you are struggling to use the system and feel that this user manual is not providing you with appropriate documentation, please contact us so that we update the document appropriatly.

# 3 Hardware overview

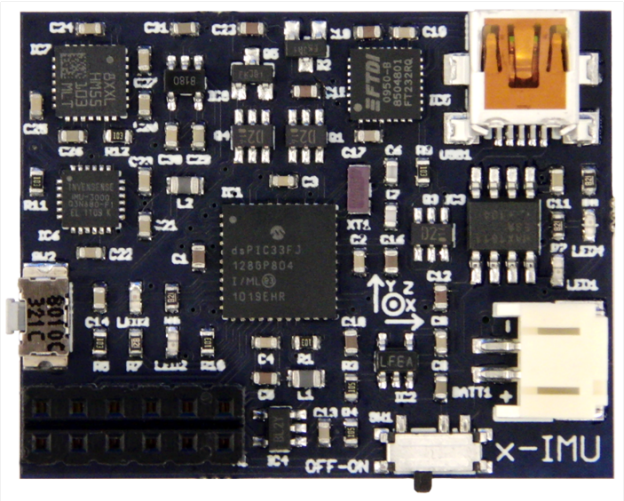



Figure 1: x-IMU top

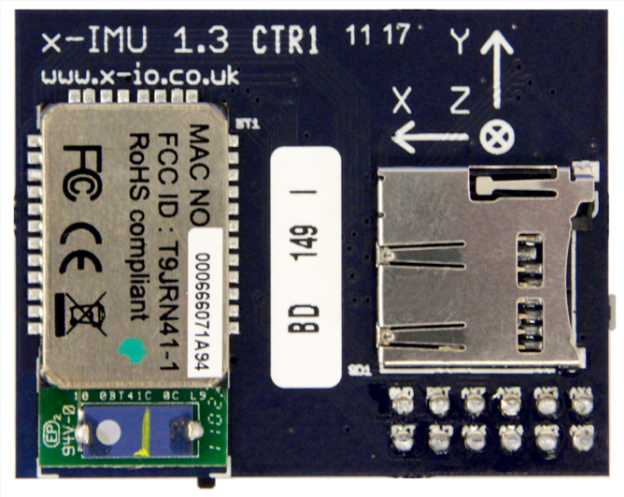



Figure 2: x-IMU bottom

## 3.1 Power switch

The power switch is used to switch the battery and USB power on or off. The battery and USB power is completely disconnected when the switch is in the *off* position. The x-IMU may be powered by an external supply (12.1) via the auxiliary port (13). For this the power switch must be in the *off* position.

## 3.2 Push button

The push button may be configured to execute x-IMU commands (15). The mode of the push button is defined in the *Button mode* register (17.2.4). The default mode is *sleep/wake*.

## 3.3 LEDs

### 3.3.1 Status LED (Green)

The green LED indicates the status of the x-IMU. The green LED will remain lit while the device is sampling and sending data packets and will otherwise be extinguished; for example, during the execution of commands (15). In sleep mode (12.4) the green LED will blink once every 3 seconds. The green LED will flash rapidly while the on-board bootloader is active.

### 3.3.2 SD card LED (Amber)

The amber LED indicates SD card (7) activity. Details on the SD LED behaviour are provided in the SD card LED section (7.2).

### 3.3.3 Bluetooth LED (Blue)

The blue LED indicates the state of the Bluetooth (6) connection and power status. Details of the Bluetooth LED behaviour are summarised in Table 2.

### 3.3.4 Charging LED (Red)

The red LED indicates the charging state of the battery (12.2). The red LED will remain lit while the battery is charging and will be extinguished once the battery is charged.

## 3.4 USB socket

The USB mini-B socket is used to connect the x-IMU to a computer via a standard USB *A to mini B (5 pin)* type cable. The USB connect can also be used to power the x-IMU. See the USB section (5) for more information.

## 3.5 Micro-SD card socket

The micro SD card socket is used to log all data generated by the x-IMU to an SD card. The x-IMU supports standard SD and HCSD cards formatted as either FAT16 or FAT32. A new file is created and logging starts each time the x-IMU is switch on, wakes up or is reset. The file is automatically closed when the x-IMU enters sleep mode (12.4) or is reset. The file must be closed before the SD card is removed or the x-IMU switched of otherwise the current file will corrupt and data lost. See the SD card section (7) for more information.

## 3.6 Bluetooth radio

The on-board Bluetooth radio is used to connect the x-IMU to a computer via Bluetooth. The on-board Bluetooth radio is a class I device with a maximum range of 100 m. The radio uses the Serial Port Profile (SPP) to enable connection to any Bluetooth host without the need to install specific drivers. Power to the Bluetooth radio can be disabled (17.2.63) to converse power. See the Bluetooth section (3.6) for more information.

## 3.7 Battery connector

The on-board battery connector allows the x-IMU to be powered by any single-cell Lithium Polymer battery. The battery is automatically charged while the x-IMU is connected via USB. See the battery and charging section (12.2) for more information.

# 4 Software overview

## 4.1 x-IMU GUI

### 4.1.1 Tab page: Serial port

The serial port tab page is used to manage the USB or Bluetooth connection between the software and the x-IMU. The USB and Bluetooth connections will each appear as a separate serial port; see the USB section (5) and Bluetooth section (6) for more information and how to find the serial port name assigned to each connection.

To connect to the x-IMU, the user first select the correct serial port name the x-IMU appears as in the *Port name* drop down list. If the name does not appear in the list, the user can either press the *Refresh List* button to update the drop down list or type in the port name directly. The *Open Port* button may then be pressed to connect to the device.

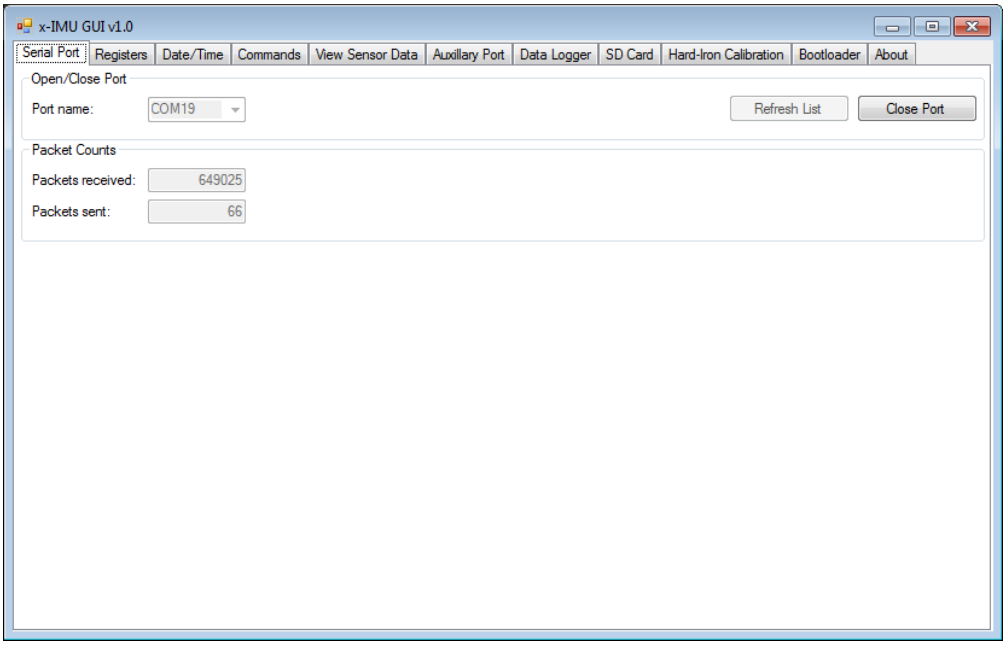



Figure 3: x-IMU GUI serial port tab page

### 4.1.2 Tab page: Registers

The registers tab page allows the user to view, edit and back up all internal settings on the x-IMU; see the registers section (17) for more infomation on x-IMU registers. All registers are organised into sections within a tree view where the end node of each branch is an individual register name and text box or drop down list containing the register value. Register values that have been read directly from the x-IMU or loaded from file will appear as blue text. Any registers values then edited will appear as red text. A right click on any register will show the action menu.

To read all register on the x-IMU, the user should right click anywhere in the registers tab page and select *Read all registers*. The software will then read each register and update the values in the tree view. Individual registers or groups of registers may be read by first selecting a register or group within the tree view and then selecting *Read this register only* or *Read all registers in this group only*. Register values in the tree view may be written to teh x-IMU using the *Write all regisers*, *Write this register only* and *Write all registers in this group only* options in the action menu.

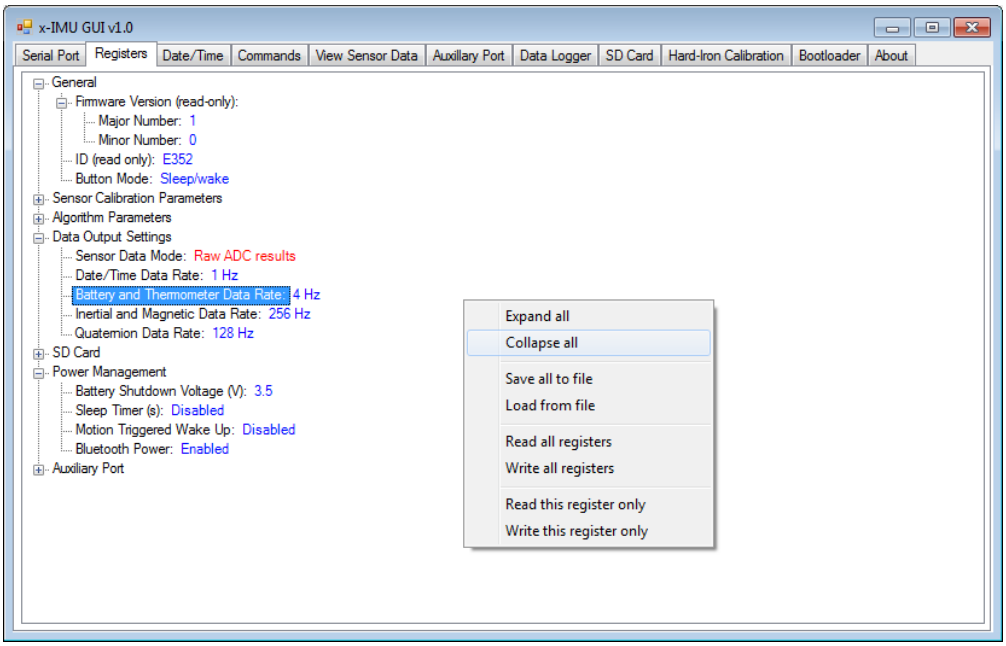



Figure 4: x-IMU GUI registers tab page with (right click) action menu

### 4.1.3 Tab page: Date/time

The date/time tab page allows the user to view and set the date and time of the x-IMU's real-time clock and calendar. The *Received date/time* text box displays the date and time each time it is received from the x-IMU. The *Read Date/Time* button may be used to read the current date and time of the x-IMU; this is of use if date/time data rate has been disabled. Pressing the *Set Date/Time* button will set the x-IMU date and time equal to computer date and time.

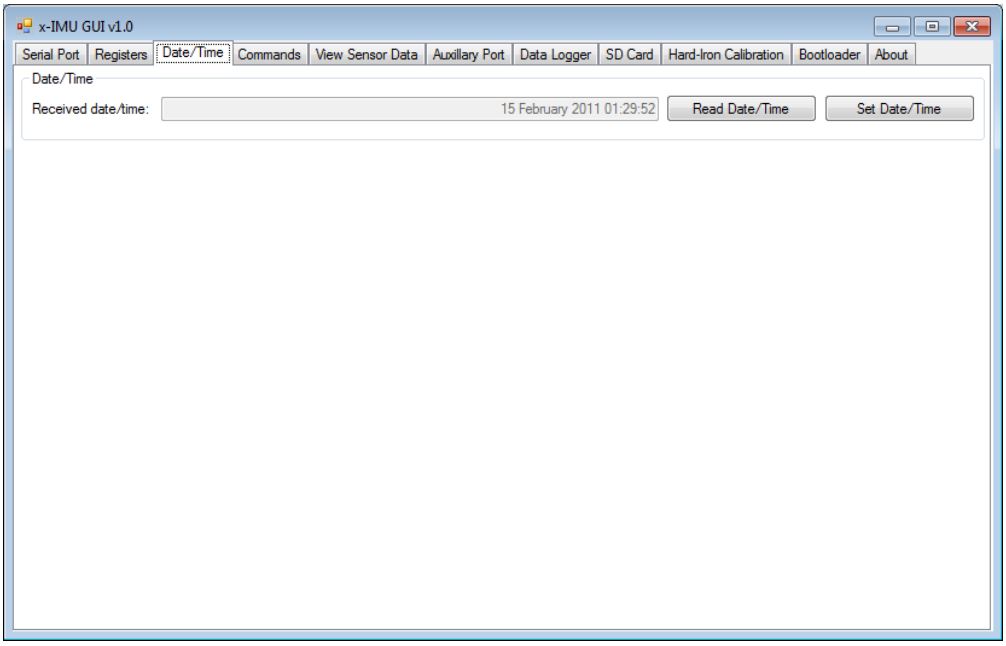



Figure 5: x-IMU GUI date/time tab page

### 4.1.4 Tab page: Commands

The commands tab page is used to send commands to the x-IMU. See the commands section (15) for more information on individual commands. Once the x-IMU has processed a command it will echo the command back and it will appear in a message box. To suppress these message boxes, un-check the *Display received command messages in message box* check box.

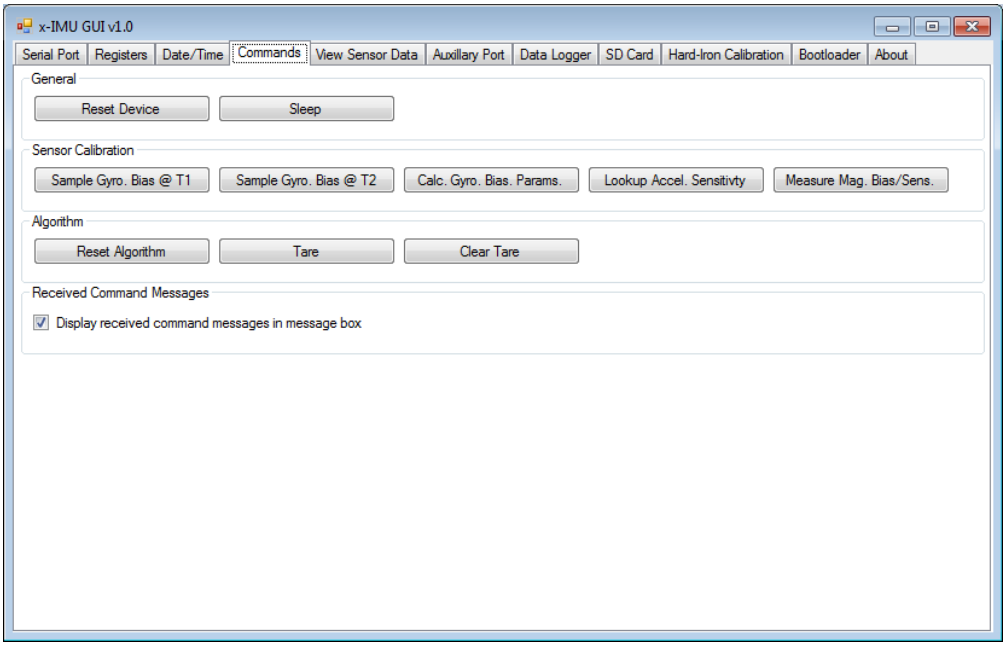



Figure 6: x-IMU GUI date/time tab page

### 4.1.5 Tab page: View sensor data

The view sensor data tab page contains buttons to show or hide separate real-time data graphic windows for incoming x-IMU sensor data.

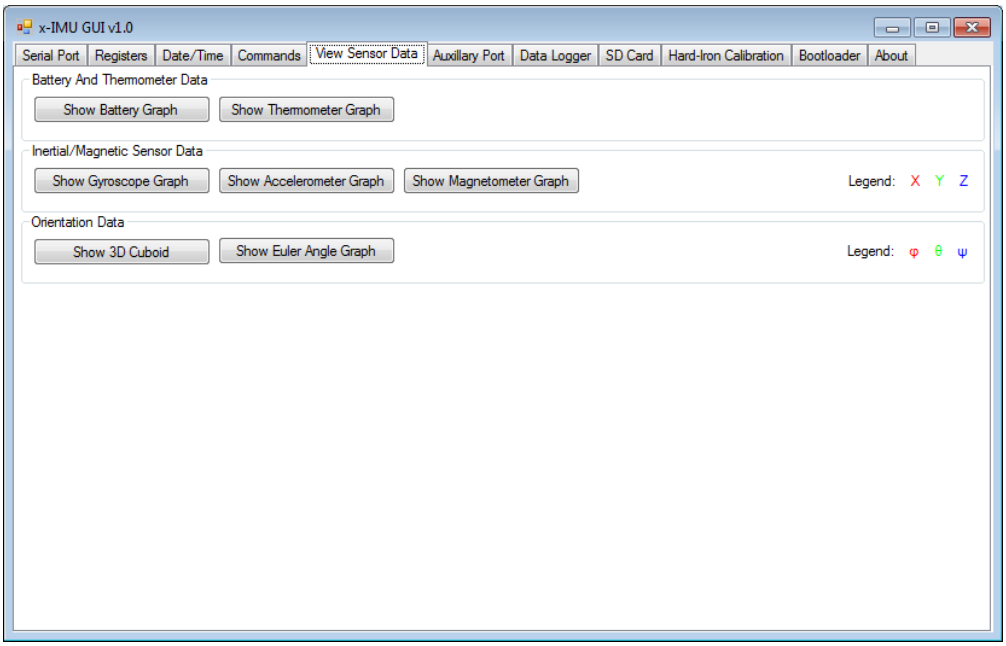



Figure 7: x-IMU GUI view sensor data tab page

The data from individual sensors is displayed in real-time data graphs as seen in Figure 7. The controls bar at the bottom of each graph allow the view and scaling to be adjusted.

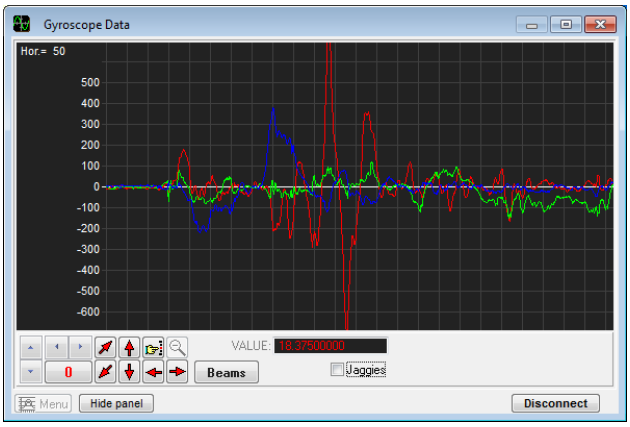



Figure 8: x-IMU GUI gyroscope data window

Orientation data received may be displayed in a graph as ZYX Euler angles and displayed as the orientation of a 3D cuboid as seen in figure 9. The cuboid is displayed in a screen coordinate frame where the x-axis is aligned to the width of the screen (left to right), the z-axis aligned to the height (bottom to top) and the y-axis projects into the screen. To align the motion of the physical x-IMU and 3D cuboid displayed on the screen, the user should first align the axes of the physical x-IMU to the screen coordinate frame and then use the *Tare* command (**??**).

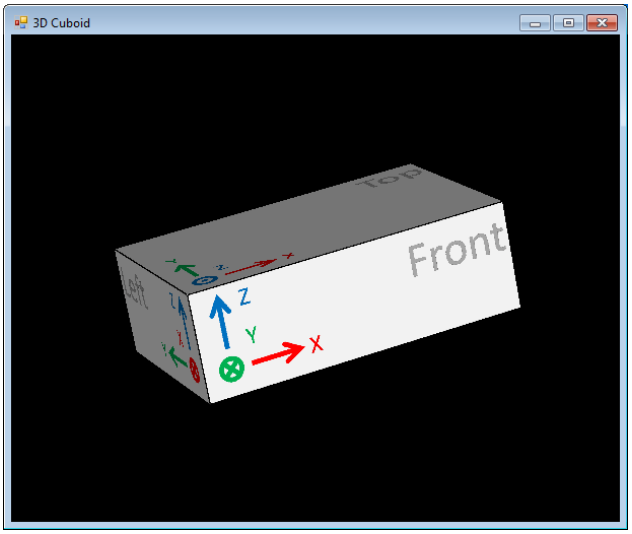



Figure 9: x-IMU GUI 3D cuboid window

#### 4.1.6 Tab page: Auxillary port

The auxiliary port tab page contains buttons to show or hide individual control windows for the different modes of the auxiliary port.

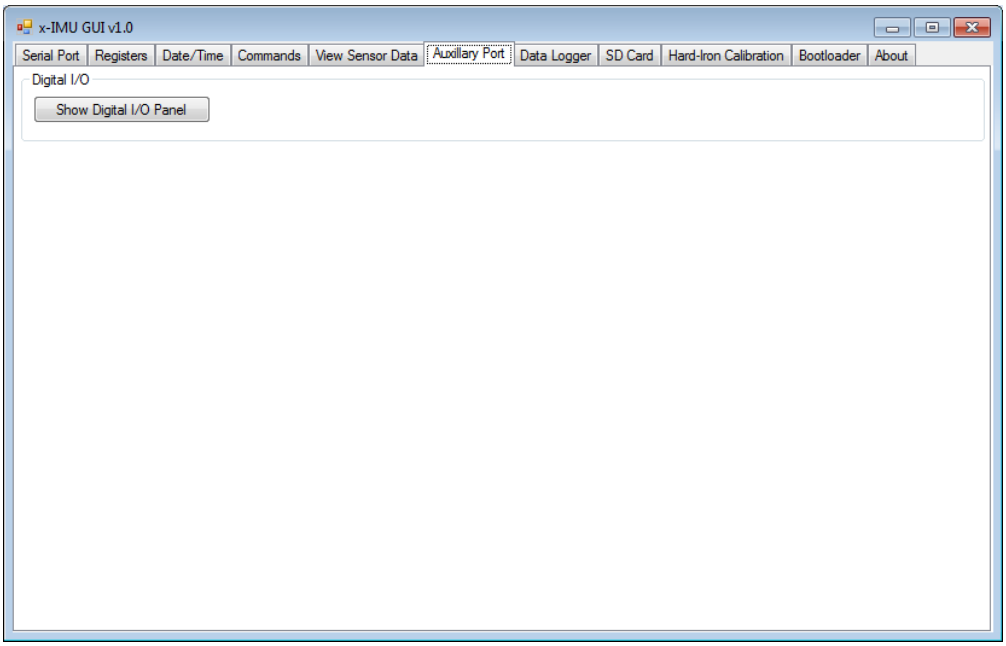



Figure 10: x-IMU GUI auxiliary port tab page

**Digital I/O control panel** The digital I/O control panel displays the state and mode of each channel of the auxiliary port when in digital I/O mode as shown in figure 11. Each channel is represented by a check box. If the channel mode is output then the check box is enabled and may be checked or un-checked to set the channel high or low respectively. If the channel is an input the check box is disabled and will be checked or un-checked if the channel is high or low respectively.

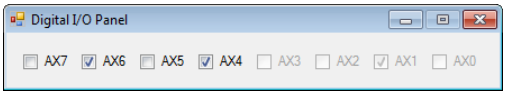



Figure 11: x-IMU GUI digital I/O control panel

### 4.1.7 Tab page: Data logger

The data logger tab allows the user to log incoming real-time data to file. These files may be imported to user software such as Microsoft Excel and MATLAB. The user may select the location and first part of the file name in the *File path* text box. This file name will be extended with an appropriate description and extension when the individual data files are created. For example, if a file name of `myFile` is specified, Euler angle and date/time data will be saved to `myFile_EulerAngles.csv` and `myFile_DateTime.txt`.

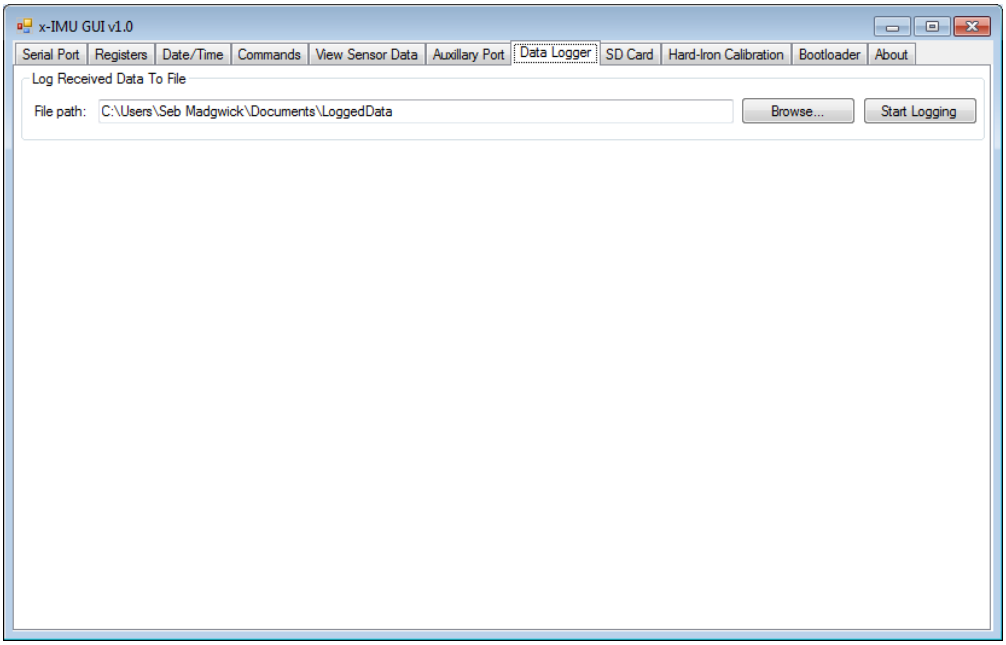



Figure 12: x-IMU GUI data logger tab page

The *Start/Stop Logging* button is used to start and stop the data logger. When logging is stopped, a report window will be presented detailing the number of each type of packet logged and the specific data files created; as shown in figure 13.

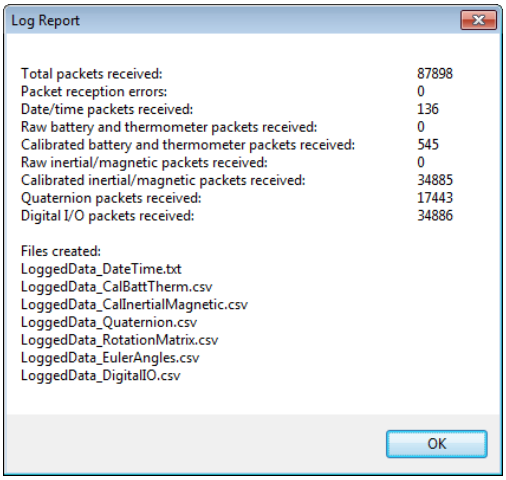



Figure 13: x-IMU GUI data logger report

### 4.1.8 Tab page: SD card

The SD card tab page allows the user to convert binary files (`.bin`) saved to the SD card in to readable data files. These files may be imported to user software such as Microsoft Excel and MATLAB. The location and file name must be specified in the *File path* text box. The file conversion will start when the *Convert* button is clicked. This process occurs in the background and may take a while if a large binary file is specified.

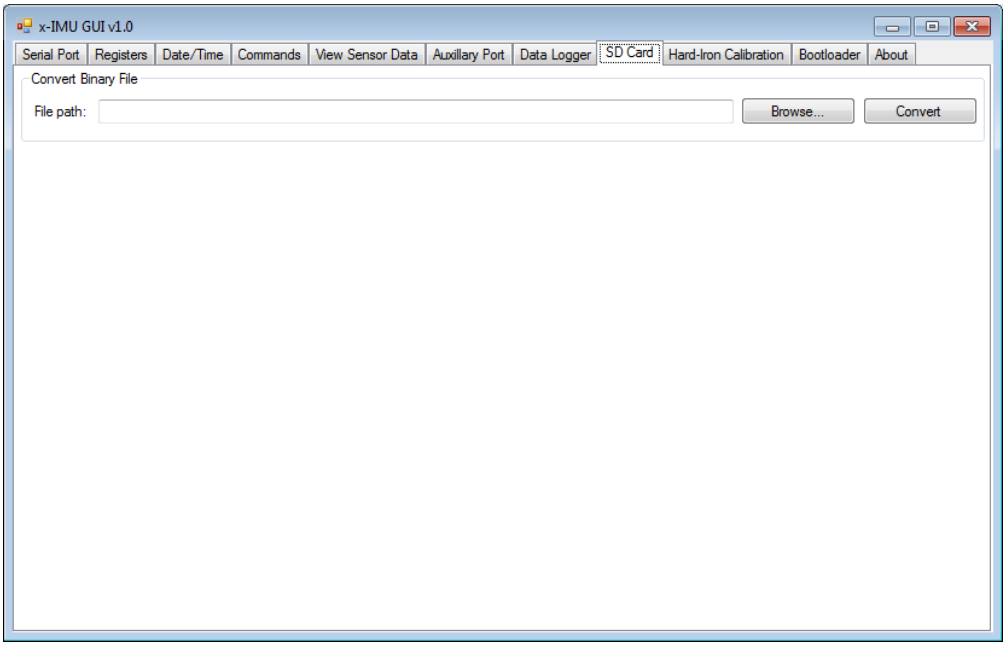



Figure 14: x-IMU GUI SD card tab page

Once the conversion is complete, a report window will be presented detailing the number of each type of packet read and the specific data files created; as shown in figure 15.

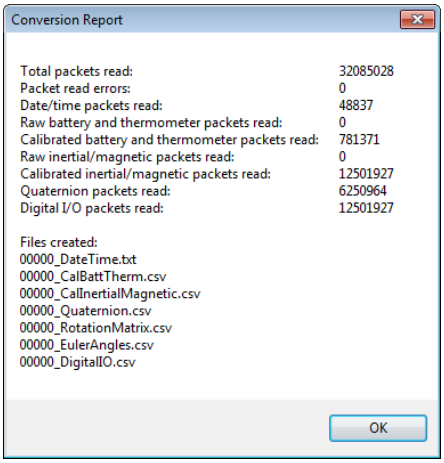



Figure 15: x-IMU GUI binary file conversion report

### 4.1.9 Tab page: Hard-iron calibration

The hard-iron calibration tab page provides all the functionality required for the user to calibrate for hard-iron interferences affecting the x-IMU. It is necessary to re-calibrate hard-iron parameters whenever the x-IMU's magnetic characterises are changed; for example, when the x-IMU if fitted to a battery or mounting that includes ferromagnetic elements. The 3 group boxes, *Step 1 - Clear Hard-Iron Bias Registers*, *Step 2 - Collect Hard-Iron Calibration Dataset* and *Step 3 - Run Hard-Iron Calibration Algorithm* represent the 3 steps that must be performed in order. See the hard-iron calibration section (**??**) for more information.

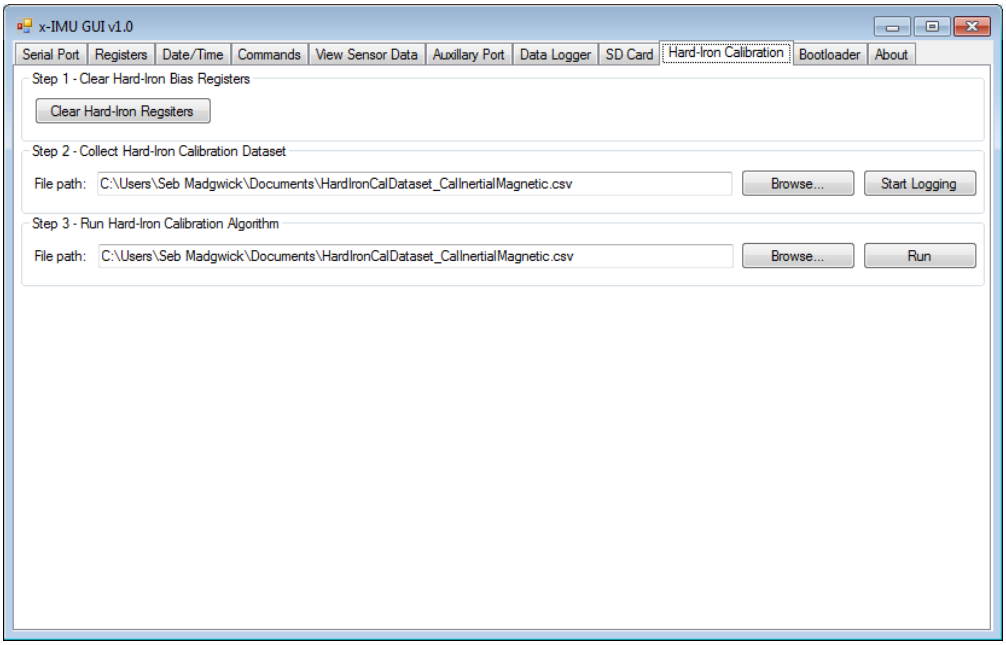



Figure 16: x-IMU GUI hard-iron calibration tab page

### 4.1.10 Tab page: Upload firmware

The bootloader tab page allows the x-IMU firmware to be updated. The bootloader may only be used while the x-IMU is connected via USB and the serial port closed. See the bootloader section (**??**) for more information.

### 4.2 x-IMU API

The x-IMU API (Application Programming Interface) is a code library that contains all the classes, data structures and methods required to interface to all features and functionality of the x-IMU. The x-IMU API is an open source project written in C# and targets Microsoft .NET 3.5. Documentation for use of the API is represented by the XML comments throughout the source code which is accessed automatically by Visual Studio's IntelliSense. The open source x-IMU GUI serves as a comprehensive template for use of all features of the x-IMU API. See the x-IMU Examples web page for further open-source examples and applications.

## 5 USB

The x-IMU streams all communication data simultaneously and identically via USB, Bluetooth (6) and to a file on the SD card (7). The USB and Bluetooth connections are also be used to send commands (15), read/write registers and settings (17) and control the auxiliary port outputs (13) from the host computer. As both USB and Bluetooth connections appear as serial ports, use of either communication channel is identical.

The x-IMU can be connected to a computer via a standard USB *A to mini B (5 pin)* type cable. The on-board FTDI USB chip is widely used USB interface with drivers available for Windows, Mac OS X and Linux. Once the drivers have been installed (5.1) and the x-IMU connected to the computer, the x-IMU will appear as a serial port and be assigned an available port name; for example *COM2*. The computer may then communicate with the x-IMU by opening this serial port. This is achieved via the *Serial Port* tab page (4.1.1) of the x-IMU GUI.

The USB connection is a reliable communication channel that cannot be comprised by user settings; the x-IMU will not enter *sleep mode* (12.4) due to the *sleep timer* (12.5) or *low battery voltage detection* (12.3) while the USB is connected. The USB connection can be used to power the x-IMU and is used by the on-board charging circuit to charge the battery if connected (12.2). The on-board USB interface is powered directly by the USB connection so that the x-IMU will remain detectable and the serial port may be held open by the computer even while the x-IMU is switched off or in sleep mode (12.4).

### 5.1 Installing USB drivers

The Windows USB drivers can be downloaded from the x-IMU webpage. Drivers for other operating systems are available of the FTDI website. To install the Windows drivers, simply run the `.exe` file. This will automatically detect specific Windows operating system being used and install the correct drivers. Once the drivers have been installed and the x-IMU connected to the computer, the x-IMU will appear as a serial port and be assigned an available port name; for example *COM2*. The port name assigned to the x-IMU USB connection can be confirmed at any time by viewing the computer's *Ports* in Windows device manager; as shown in Figure 20.

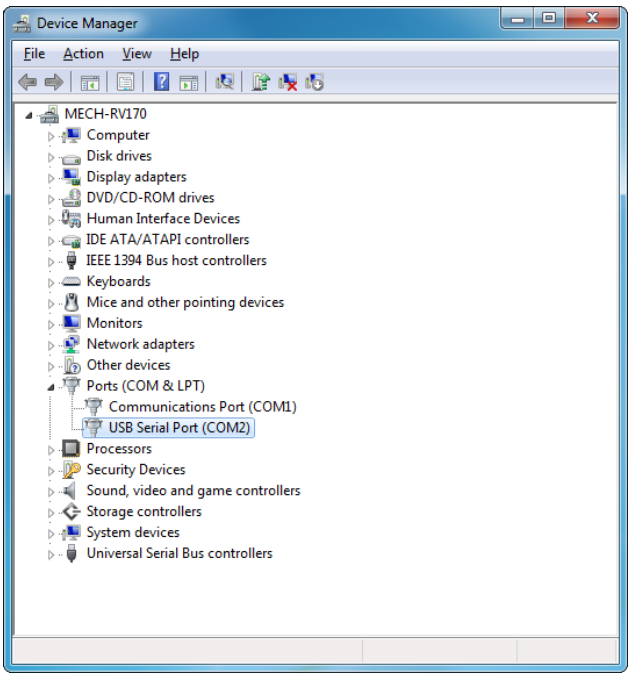



Figure 17: Confirming the port name assigned to the x-IMU USB connection

**Windows serial mouse bug** Windows may misinterpret the constant stream of data from the x-IMU as the behaviour of a serial mouse when the x-IMU USB is connected. This will lead to the mouse cursor being 'hi-jacked' by apparent random behaviour. If this happens the x-IMU should be unplugged and reconnected while switched off or in sleep mode (15.1.3) for the first few seconds of connection. The 'hi-jacked' activity may leave the mouse buttons disabled which can be undone by entering and then leaving the *Ctrl + Alt + Del* screen.

### 5.2 USB bandwidth

It is possible for the user to define data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58) so that the data being generated by the x-IMU exceeds the bandwidth of a communication channel. If the USB bandwidth is exceed, the USB transmit buffer will overrun and some data will be lost. When this happens a *USB transmit buffer overrun* error (16.1.4) will be generated. As this error message is sent immediately after the buffer has overrun, the error message will be successfully transmitted. This error can be avoided by reducing the data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58).

All data sent to the x-IMU via USB is buffered in the USB receive buffer before being processed. The time required to process the received data is dependent on the data. If data is sent to the x-IMU via USB at a rate at a rate greater than it can be processed then the receive buffer will overflow and some data will be lost. When this happens a *USB receive buffer overrun* error (16.1.3) will be generated.

## 6 Bluetooth

The x-IMU streams all communication data simultaneously and identically via USB (5), Bluetooth and to a file on the SD card (7). The USB and Bluetooth connections are also be used to send commands (15), read/write registers and settings (17) and control the auxiliary port outputs (13) from the host computer. As both USB and Bluetooth connections appear as serial ports, use of either communication channel is identical.

The on-board Bluetooth radio is a class I device with a maximum range of 100 m. The radio uses the Serial Port Profile (SPP) to enable connection to any Bluetooth host without the need to install specific

drivers. Once paired (6.1) with a host computer, the x-IMU will appear as a serial port and be assigned an available port name; for example *COM3*. The computer connects to the x-IMU via Bluetooth by opening this serial port. This is achieved via the *Serial Port* tab page (4.1.1) of the x-IMU GUI. The Bluetooth connection will be lost when the x-IMU is switch off, enters sleep mode or is out of range. The connection status of the x-IMU is indicated by the Bluetooth LED (6.2). The Bluetooth radio can be completely disabled by the user via the *Bluetooth power* register (17.2.63) to conserve power.

## 6.1 Pairing the x-IMU with a Bluetooth host

As with any Bluetooth device, the x-IMU must first be paired with the host computer before a Bluetooth connection can be made. This pairing process is the same for all Bluetooth devices and will be familiar those who have used other Bluetooth devices such as printers or mobile phones.

To pair the x-IMU with a host computer, the host computer's Bluetooth must be enabled and the x-IMU must be switched on and the Bluetooth radio powered (17.2.63). The user may then use the host computer to search for and the x-IMU to be paired with the computer. The x-IMU will appear with the name *x-IMU-0000* where the last 4 digits are the x-IMU's *Device ID* (17.2.3). For example, Figure 18 shows how this is done in Windows 7 having right clicked the Bluetooth icon the task bar.

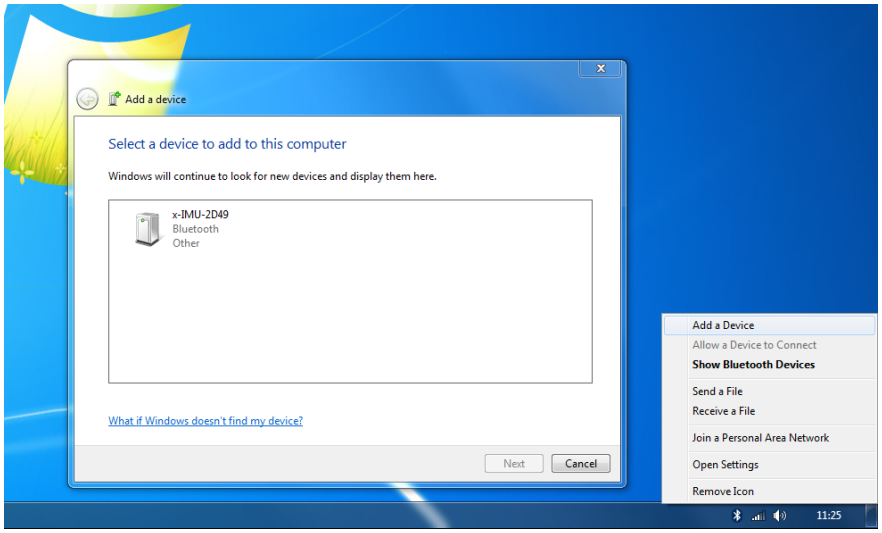



Figure 18: Searching for the x-IMU as a new Bluetooth device in Windows 7

Once the x-IMU has been found by the host computer, it can be added. This will require the user to enter the x-IMU's Bluetooth pass code: '1234'. The x-IMU Bluetooth pairing will be assigned an available serial port name by the host computer; for example *COM3*. For example, Figure 19 shows this being done in Windows 7.

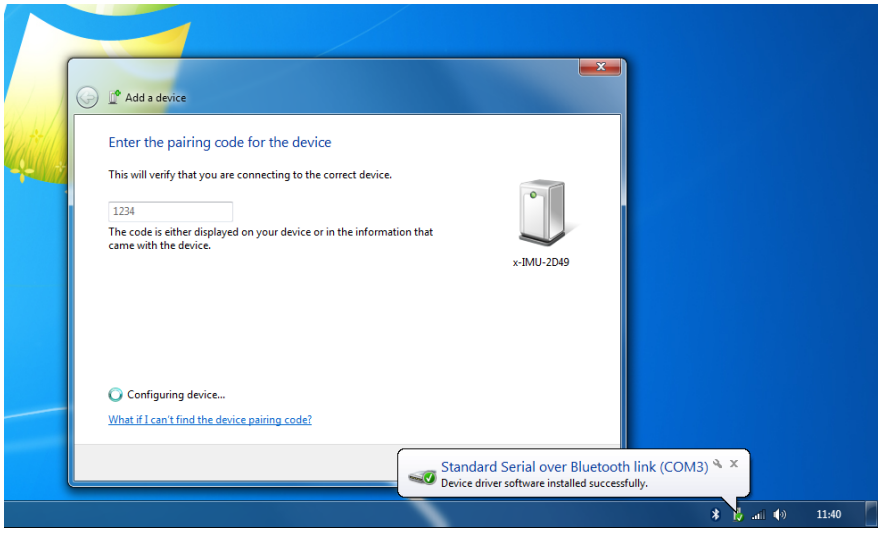

Add a device
Enter the pairing code for the device
This will verify that you are connecting to the correct device.
1234
The code is either displayed on your device or in the information that came with the device.
x-IMU-2D49
Configuring device...
What if I can't find the device pairing code?
Standard Serial over Bluetooth link (COM3)
Device driver software installed successfully.
11:40

Figure 19: Adding the x-IMU as a new Bluetooth device in Windows 7

The port name assigned to the x-IMU Bluetooth pairing can be confirmed at any time by viewing the services of the x-IMU. For example, Figure 20 shows how this is done in Windows 7 having right clicked the x-IMU Bluetooth device icon.

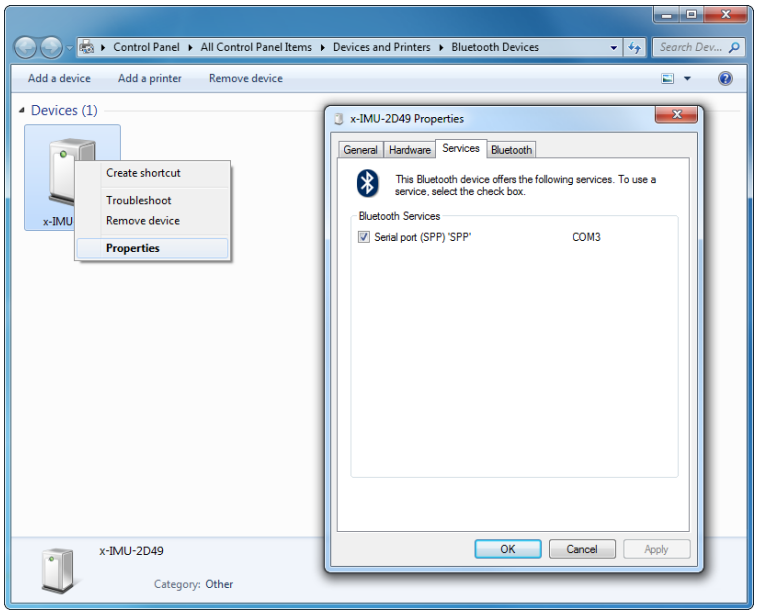



Figure 20: Confirming the port name assigned to the x-IMU Bluetooth pairing

## 6.2 Bluetooth LED

The blue Bluetooth LED indicates the Bluetooth radio state. The LED behaviour and associated Bluetooth radio states are detailed in table 2.

| LED behaviour | Bluetooth state |
|---|---|
| Off | Switched off. Power to the radio is completely disconnected |
| Flashing (1 Hz) | Fully powered and discoverable |
| On | Fully powered and connected |

Table 2: Bluetooth LED states

### 6.3 Bluetooth bandwidth

It is possible for the user to define data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58) so that the data being generated by the x-IMU exceeds the bandwidth of a communication channel. If the Bluetooth bandwidth is exceed, the Bluetooth transmit buffer will overrun and some data will be lost. When this happens a *Bluetooth transmit buffer overrun* error (16.1.6) will be generated. As this error message is sent immediately after the buffer has overrun, the error message will be successfully transmitted. This error can be avoided by reducing the data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58).

All data sent to the x-IMU via Bluetooth is buffered in the Bluetooth receive buffer before being processed. The time required to process the received data is dependent on the data. If data is sent to the x-IMU via Bluetooth at a rate at a rate greater than it can be processed then the receive buffer will overflow and some data will be lost. When this happens a *Bluetooth receive buffer overrun* error (16.1.5) will be generated.

### 6.4 Optimising Bluetooth performance

The practical range and quality of the Bluetooth connection are dependent on a number of factors. A poor Bluetooth connection will be unable to handle higher data output rates and so result in missing data and *Bluetooth transmit buffer overrun* errors (16.1.6). The use of lower data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58) can help achieve a more reliable Bluetooth communication channel.

The x-IMU uses a class I Bluetooth radio which represents a maximum range of 100 m. However, the practical performance is also limited by computer's Bluetooth class; for example a class II Bluetooth dongle (representing a range of 10 m) will limit the x-IMU's operating range to 10 m. Performance also varies between Bluetooth dongle brands; a dongle from a reputable brand may be expected to perform better than a low-cost, 'budget' product. Bluetooth is a radio system and so the location of the antennae (usually built into the dongle) should be given consideration. For example, a miniature Bluetooth dongle plugged in to the back a desktop PC can be expected to achieve worse performance than if the dongle was fixed to a front USB port with line-of-sight to the x-IMU.

## 7 SD card

The x-IMU streams all communication data simultaneously and identically via USB (5), Bluetooth (6) and to a file on the SD card. The SD card may therefore be used in conjunction with the USB and/or Bluetooth or as the sole communication channel allowing the x-IMU to function as a standalone data logger. Data is logged to the SD card on separate files binary files that are automatically created each time the x-IMU is switched on, reset or wakes up. Logging is only then stopped once the x-IMU is reset or enters sleep mode.

The binary files (`.bin`) created may be read form the SD card on to any PC and then converted to individual Comma Separated Variable (`.csv`) files using via the x-IMU GUI *SD Card* tab page (4.1.8). Alternatively the x-IMU Binary File Converter may be used for command-line-based or automated conversion of multiple files. Converted CSV and text files can be directly imported into programmes such as MATLAB and Microsoft Excel. The MATLAB x-IMU Library includes all the tools required to import, structure and plot x-IMU data.

The x-IMU supports conventional SD cards and HCSD cards. Cards may be formatted as FAT16 (usually cards equal or less than 2 GB) and FAT32 (for card greater than 2 GB). For reliable performance it is recommended that the SD card is formatted prior to each use.

## 7.1 Creating and closing files

The x-IMU automatically creates a new file on the SD card each time the x-IMU is switched on, reset or wakes up. If an SD card is not accessible at this point, the x-IMU will not create a file and the SD card will not be used. The new file name is created as the 5 digit number stored in the *SD card new file name* register (17.2.59); for example, `00000.bin`. The number stored in this register is automatically incremented each time a new file is created. This ensures that each file created by the x-IMU is given a unique file name until the maximum file name of `65535.bin` is reached, the file name will then automatically reset to `00000.bin` and start again. The user can reset this value to any number by writing to the register (17.2.59). If the x-IMU attempts to create a file name that already exists on the SD card, the x-IMU automatically increment the file name and try again. If all file names have been used, the x-IMU will not create a file and the SD card will not be used.

File must be closed before the SD card is removed or the x-IMU switched off otherwise the file will be corrupted and all data written to the file will have been lost. The file is automatically closed when the x-IMU is reset or enters sleep mode (12.4). Users wishing to frequently remove the SD card may wish to have the push button (3.2) configured in *sleep/wake* mode.

## 7.2 SD card LED

The amber SD card LED indicates SD card activity. The LED remains lit each time a burst of data is written to the SD card. If the user low defines data output rates then the LED will blink infrequenctly, high data output rates will mean the LED will flash rapidly. The SD card LED therefore provides an indication of SD card bandwidth performance (7.3).

## 7.3 SD card bandwidth

It is possible for the user to define data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58) so that the data being generated by the x-IMU exceeds the bandwidth of a communication channel. The SD card bandwidth is greater than the USB and Bluetooth bandwidth and so the SD card may still provide reliable data logging when the USB or Bluetooth channel bandwidth is exceeded. If the SD card bandwidth is exceed, the SD card buffer will overrun and some data will be lost. When this happens an *SD card write buffer overrun* error (16.1.7) will be generated. As this error packet is sent immediately after the buffer has overrun, the error message will be successfully logged to the SD card. This error can be avoided by reducing the data output rate settings (e.g. 17.2.56, 17.2.57, 17.2.58). The SD card LED (7.2) may be used to provides an indication of SD card bandwidth performance while access to error messages is not available.

The effective bandwidth of SD card varies between different SD card manufacturers and may decrease significantly if the SD card becomes fragmented. It is therefore recommended that the SD card is formatted prior to each use.

## 7.4 Magnetic distortions from the SD card socket

The SD card socket contains a ferromagnetic mechanism that may distort magnetometer measurements in different ways dependant of the presence of an SD card. These distortions are removed through hard-iron calibration and each x-IMU is calibrated and supplied with a dummy SD card that may be used to ensure constant SD card socket magnetic characteristics.

# 8 Real-time clock and calendar

The on-board real-time clock and calendar provides accurate measurement of the date and time and is pre-programmed to account for leap-years between the year 2000 and 2099. The real-time clock and calendar data can be viewed and synchronised with the computer clock using the x-IMU GUI via the *Date/Time* tab page (4.1.3).

The real-time clock and calendar data a is provided by the x-IMU in the *write date/time data* packets. The output rate of these packets of either *disabled* or 1 Hz is specified in the *date/time data rate* register

(17.2.55). A single *date/time data* packet is always sent on device reset regardless of user settings so that the date and time are hardcoded as the first packet written to the SD card (7). The real-time clock and calendar is set by sending a *write date/time data* packet to the x-IMU, once the new date and time have been set the x-IMU will respond with a *write date/time data* containing the real-time clock and calendar data. The date and time may read at any time by sending a *read date/time data* packet to the x-IMU.

### 8.1 Maintaining clock power

The real-time clock and calendar require power to operate. If power is lost or the x-IMU switch off then the date and time will reset to 01/01/2000 00:00:00. Applications that require date and time to be maintained should ensure that the x-IMU is never switched off and instead take advantage of sleep mode (12.4).

## 9 Sensors

The x-IMU's on-board sensors include a triple axis gyroscope, triple axis accelerometer, triple axis magnetometer, thermometer and a battery voltmeter. The user may access individual sensor data as either *Raw un-calibrated ADC results* or as *Calibrated units* by specifying the mode in the *Sensor data mode* register (17.2.54). The data from individual sensors is provided in either the *Raw inertial/magnetic data* and *Raw battery and thermometer data* packets or the *Calibrated inertial/magnetic data* and *Calibrated battery and thermometer data* packets. The data output rate of these packets may be specified up to 256 Hz in the *Battery and thermometer data output rate* (17.2.56) and *Inertial/magnetic data output rate* (17.2.57) registers. Alternatively the individual output rates may be *disabled* to prevent data from being sent.

### 9.1 Battery voltmeter

The battery voltmeter allows the battery voltage to be monitored by the user application. The battery voltmeter must be correctly calibrated if the low battery voltage detection functionality is to be used. The battery voltmeter has 12-bit resolution and a range of 0 V to 6.6 V. When the power switch is in the *off* position (so that the x-IMU may be powered from an external source via the auxiliary port) the battery voltmeter will instead measure the voltage of the external source.

**Raw ADC data:** In raw data mode the battery voltmeter data is the ADC integer value between 0 and 4096 corresponding to a voltage between 0 V and 6.6 V. This data is provided in the *Raw battery and thermometer data* packets.

**Calibrated data:** In calibrated data mode the battery voltmeter data is the calibrated measurement in Volts. This data is provided in the *Calibrate battery and thermometer data* packets. The calibrated measurement $v$ is calculated from the raw ADC measurements $\tilde{v}$ according to a sensitivity $s_v$ and bias $b_v$ as described by equation (1). Parameters $b_v$ and $s_v$ are defined in the battery voltmeter sensitivity (17.2.5) and bias (17.2.6) registers.

$$v = \frac{1}{s_v}(\tilde{v} - b_v) \tag{1}$$

### 9.2 Thermometer

The thermometer is built in to the gyroscope and provides a measurement of the temperature of the device. The thermometer must be correctly calibrated for calibrated gyroscope measurements to compensate for gyroscope bias temperature sensitivity. The thermometer has 16-bit resolution and has a range of -30°C to +85°C. See the IMU-3000[1] datasheet for further information on the thermometer's characteristics.

[1] x-IMUs of generations prior to version 1.3 use the ITG-3200 gyroscope.

**Raw ADC data:** In raw data mode the thermometer data is the ADC integer value between $-32,768$ and $+32,767$ linearly proportional to temperature. This data is provided in the *Raw battery and thermometer data* packets.

**Calibrated data:** In calibrated data mode the thermometer data is the calibrated temperature in °C. This data is provided in the *Calibrate battery and thermometer data* packets. The calibrated measurement $\tau$ is calculated from the raw ADC measurement $\tilde{\tau}$ according to a defined sensitivity $s_\tau$ and bias $b_\tau$ as described by equation (2). Parameters $b_\tau$ and $s_\tau$ are defined in the battery voltmeter sensitivity (17.2.7) and bias (17.2.8) registers.

$$\tau = \frac{1}{s_\tau}(\tilde{\tau} - b_\tau) \tag{2}$$

## 9.3 Gyroscope

The triple axis gyroscope provides a measurement of the angular velocities around the $x$, $y$ and $z$ axes of the x-IMU. The gyroscope must be correctly calibrated in order for the IMU and AHRS algorithms to be able to function correctly; the algorithms use measurements of angular velocities to filter out errors in the estimated orientation caused by linear accelerations and temporal magnetic distortions. The gyroscope has 16-bit resolution and a range of ±2000°/s. See the IMU-3000[2] datasheet for further information on the gyroscope's characteristics.

**Raw ADC data:** In raw data mode the gyroscope data is the ADC integer values between $-32,768$ and $+32,767$ linearly proportional to angular velocities. This data is provided in the *Raw inertial/magnetic data* packets.

**Calibrated data:** In calibrated data mode the gyroscope data are calibrated angular velocities in °/s. This data is provided in the *Calibrated inertial/magnetic data* packets. The calibrated measurements $g_x$, $g_y$ and $g_z$ are calculated from the raw ADC measurements $\tilde{g}_x$, $\tilde{g}_y$ and $\tilde{g}_z$ according to the defined sensitivities $s_{g_x}$, $s_{g_y}$ and $s_{g_z}$, temperature of the device $\tau$, biases at 25°C $b_{g_x}$, $b_{g_y}$ and $b_{g_z}$ and bias temperature sensitivities $f_x$, $f_y$ and $f_z$. The calibrated measurmndts are described by equation (3). Parameters $s_{g_x}$, $s_{g_y}$, $s_{g_z}$, $b_{g_x}$, $b_{g_y}$, $b_{g_z}$, $f_x$, $f_y$ and $f_z$ are defined in the separate gyroscope calibration parameter registers (17.2.9 to 17.2.17).

$$\begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = \begin{bmatrix} s_{g_x} & 0 & 0 \\ 0 & s_{g_y} & 0 \\ 0 & 0 & s_{g_z} \end{bmatrix}^{-1} \left( \begin{bmatrix} \tilde{g}_x \\ \tilde{g}_y \\ \tilde{g}_z \end{bmatrix} - \begin{bmatrix} b_{g_x} \\ b_{g_y} \\ b_{g_z} \end{bmatrix} - \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & f_z \end{bmatrix}^{-1} \begin{bmatrix} \tau - 25 \\ \tau - 25 \\ \tau - 25 \end{bmatrix} \right) \tag{3}$$

## 9.4 Accelerometer

The triple axis accelerometer and provides a measurement of the accelerations along the $x$, $y$ and $z$ axes of the x-IMU. The accelerometer must be correctly calibrated in order for the IMU and AHRS algorithms to be able to function correctly; the algorithms use the accelerometer to measure the direction of gravity and provide an absolute reference for the pitch and roll components of the estimated orientation. The acceleroemter has 12-bit resolution and selectable ranges from ±2 g to ±8 g. The measurement range of the accelerometer is defined in *Accelerometer full scale* register (17.2.26). See the LSM303DLH datasheet for further information on the accelerometer's characteristics.

**Raw ADC data:** In raw data mode the accelerometer data is the ADC integer values between $-4096$ and $+4095$ linearly proportional to accelerations. This data is provided in the *Raw inertial/magnetic data* packets.

[2] x-IMUs of generations prior to version 1.3 use the ITG-3200 gyroscope.

**Calibrated data:** In calibrated data mode the accelerometer data are calibrated accelerations in $g$. This data is provided in the *Calibrated inertial/magnetic data* packets. The calibrated measurements $a_x$, $a_y$ and $a_z$ is calculated from the raw ADC measurements $\tilde{a}_x$, $\tilde{a}_y$ and $\tilde{a}_z$ according to the defined sensitivities $s_{a_x}$, $s_{a_y}$ and $s_{a_z}$ and biases $b_{a_x}$, $b_{a_y}$ and $b_{a_z}$ as described by equation (4). Parameters $s_{a_x}$, $s_{a_y}$, $s_{a_z}$, $b_{a_x}$, $b_{a_y}$ and $b_{a_z}$ are defined in the separate accelerometer calibration parameter registers (17.2.27 to 17.2.32). The sensitivities and biases will be different for each full-scale measurement range (17.2.26).

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} s_{a_x} & 0 & 0 \\ 0 & s_{a_y} & 0 \\ 0 & 0 & s_{a_z} \end{bmatrix}^{-1} \left( \begin{bmatrix} \tilde{a}_x \\ \tilde{a}_y \\ \tilde{a}_z \end{bmatrix} - \begin{bmatrix} b_{a_x} \\ b_{a_y} \\ b_{a_z} \end{bmatrix} \right) \tag{4}$$

### 9.5 Magnetometer

The triple axis magnetometer and provides a measurement of the magnetic flux along the $x$, $y$ and $z$ axes. The magnetometer must be correctly calibrated in order for the AHRS algorithm to be able to function correctly; the algorithm uses the magnetometer to measure the Earth's magnetic field and provide an absolute reference for the heading component of the estimated orientation. The magnetometer has 12-bit resolution and selectable ranges from ±1.3 G to ±8.1 G. The measurement range of the magnetometer is defined in *Magnetometer full scale* register (17.2.33). See the LSM303DLH datasheet for further information on the magnetometer's characteristics.

**Raw ADC data:** In raw data mode the magnetometer data is the ADC integer values between −4096 and +4095 linearly proportional to magnetic flux. This data is provided in the *Raw inertial/magnetic data* packets. A value fo -4096 will be provided when the measurement saturates in either direction.

**Calibrated data:** In calibrated data mode the magnetometer data are calibrated accelerations in $G$. This data is provided in the *Calibrated inertial/magnetic data packets*. The calibrated measurements $m_x$, $m_y$ and $m_z$ are calculated from the raw ADC measurements $\tilde{m}_x$, $\tilde{m}_y$ and $\tilde{m}_z$ according to the defined sensitivities $s_{m_x}$, $s_{m_y}$ and $s_{m_z}$, biases $b_{m_x}$, $b_{m_y}$ and $b_{m_z}$ and hard-iron biases $h_x$, $h_y$ and $h_z$ as described by equation (5). Parameters $s_{m_x}$, $s_{m_y}$, $s_{m_z}$, $b_{m_x}$, $b_{m_y}$, $b_{m_z}$, $h_x$, $h_y$ and $h_z$ are defined in the separate magnetometer calibration parameter registers (17.2.34 to 17.2.42). The sensitivities and biases will be different for each full-scale measurement range (17.2.33).

$$\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} s_{m_x} & 0 & 0 \\ 0 & s_{m_y} & 0 \\ 0 & 0 & s_{m_z} \end{bmatrix}^{-1} \left( \begin{bmatrix} \tilde{m}_x \\ \tilde{m}_y \\ \tilde{m}_z \end{bmatrix} - \begin{bmatrix} b_{m_x} \\ b_{m_y} \\ b_{m_z} \end{bmatrix} \right) - \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} \tag{5}$$

## 10 Sensor calibration

This section is currently unavailable but can be updated on request.

## 11 IMU/AHRS algorithms

- On-board algorithms provde oreitnation relative to Earth - Uses gyro, accel, mag - Oreintation data provided as quaternion in quaternion packet. Data output rate set by register - API includes class methods for conversion to rotation matrix and Euler angles. - Oreintation viewed in GUI with 3D cuboid and Euler angles.

- Algorithm update rate = 256 Hz regardless of data output rate - Algorithm performance is dependant on calibration. - Performance can be tuned by adjusting algorithm paramters. See section 11.4 for more information.

## 11.1 Principle of operation

- use gyroscope data to estimate oreintaiton.... however gyroscope measeutnts alone cannot work because of drift. - this drift is compensated for through meaurments of the Earth's gravitational and magentic feilds, provided by the accelermeter and magnetoemter respectivly. - accelerometer measures accelerations and the direction of gravity.... it is used to counter drift relative to the Earth's surface; i.e. the estimated pitch and roll. As the acceleoemter also measures accelerations as well as gravitry, - magnetometer measures direction fo Earth's magentic feild... used to counter drift relative to the magnetic poles of the Earth; i.e. in the estimated heading. - the proportional gain governs how much influence

**IMU mode:** - The IMU algorithm uses only the gyroscope and acceleormeter and so will not compensate for drift in heading. This is suitable for many applciations where only pitch and roll are required and/or it is possible to 'reset' heading periodically. - The advantage of excluding the magnetoemter from the algorithm is: magnetic calbration unessary, local magnetic distortion have no effect; e.g. if x-IMU used within a magnetically noisey enviroment

**AHRS mode:** - The IMU algorithm uses only the gyroscope and acceleormeter and magnetoemer. Free form drift in all axes. As magnetoem

## 11.2 Algorithm parameters

### 11.2.1 Algorithm mode

### 11.2.2 Algorithm gains

### 11.2.3 Algorithm magnetic feild rejection

### 11.2.4 Algorithm tare quaternion

## 11.3 Algorithm operations

### 11.3.1 Algorithm initialise

- init on reset

### 11.3.2 Algorithm tare

### 11.3.3 Algorithm clear tare

### 11.3.4 Algorithm initialise then tare

## 11.4 Optimising algorithm performance

# 12 Power management

The x-IMU may be powered via USB (5), an external supply (12.1) or a single cell lithium polyer (LiPo) battery cell (12.2).

## 12.1 External supply

The x-IMU may be powered by a 3.5 to 7.7 V external supply via the auxiliary port (13). The supply should be connected to the *GND* and *EXT* pins. This power supply is only enabled while the power switch (3.1) is in the *off* position. In this situation, the battery volt meter (9.1) will measure the voltage of the external supply.

## 12.2 Battery and charging

The x-IMU has a standard connector for a 3.7 V single cell Lithium Polymer (LiPo) battery cell. These batteries are widely available in range of capacities, for example 1000 mAh and 2000 mAh. The battery life is dependent on user settings and usage. See the tips on minimising power consumption section (12.7).

The x-IMU has an on-board battery charger specially designed for LiPo battery cells. The battery is charged automatically while the x-IMU is connected to a USB port. The red charging LED (3.3.4) will remain lit while the battery is charging. Charging stops automatically once complete. The x-IMU may be used as normal while the battery is charging. It is not necessary for the connected computer to have the USB drivers installed for charging; however, the charging process will be faster if the drivers are installed.

## 12.3 Low battery voltage detection

The calibrated battery voltmeter (9.1) is used to trigger sleep mode (12.4) once the battery voltage falls below a specific level defined in the *battery shutdown voltage* register (17.2.60). This allows the x-IMU to execute critical tasks prior to power failure; for example closing the file on the SD card and notifying the user or host software of the situation (16.1.2). By entering sleep mode prior to power failure, the x-IMU also ensures that the date and time of the real-time clock and calendar (8) are not lost. The low battery voltage detection is disabled while USB is connected.

## 12.4 Sleep mode

In sleep mode, the x-IMU remains powered but all on-board components are shutdown. This allows the device to be effectively 'turned off' without disabling power to essential components; for example, the real time clock and calendar (8). The green status LED (3.3.1) will blink once every 3 seconds to indicate that the device is in sleep mode. Sleep mode is enabled through the sources listed below. The x-IMU will reset upon wake up so that the same behaviour may be expected when the devices is powered on, reset or awakened. The wake up sources are listed below.

**Sleep mode enable sources:**

- Push button (3.2) in *sleep/wake* mode
- Sleep command via USB or Bluetooth
- Low battery voltage detection (12.3)
- Sleep timer (12.5)

**Wake up sources**

- Push button (3.2) in sleep/wake mode
- Motion trigger wake up (12.6)

## 12.5 Sleep timer

The sleep timer will trigger sleep mode after the period of time defined in the *sleep timer* register (17.2.61) has elapsed. The sleep timer countdown starts when the x-IMU starts up and may be reset by the sources listed below. These sources enable the detection of motion, the user or the host software to prevent the x-IMU from entering sleep mode. The sleep timer is disabled by specifying a *sleep timer* register (17.2.61) value of 0 seconds. The sleep timer is disabled while USB is connected.

**Sleep timer reset sources**

- *Reset sleep timer* command (15.1.4)
- Motion trigger wake up (12.6)

### 12.6 Motion triggered wake up

The motion trigger wake up is enabled via the *motion trigger wake up* register (17.2.62) and may be either disabled or set to a *low* or high *sensitivity*. Motion is detected using accelerometer. If motion is detected while the x-IMU is in sleep mode then the x-IMU will wake up. While the x-IMU is not in sleep mode the motion trigger wake up is used to reset the sleep timer (17.2.61) and thus postpone sleep while motion persists. For example, if the sleep timer is set to 20 seconds and there is sufficient motion once every 10 seconds the motion trigger wake up will prevent the sleep timer from expiring.

### 12.7 Tips for minimising power consumption

Battery powered applications require that power consumption is minimised in order to extend the battery life. The x-IMU is designed to optimised power consumption according to user settings. The user may therefore expect a considerable reduction in power consumption and extended battery life simply by using register settings appropriate to their application.

#### Tips

- Set data rates of unused data to *Disabled*.
- Use the minimum data rates required by application.
- Set IMU/AHRS algorithm (11) mode to *Disabled* if not required.
- Disable Bluetooth (6) if not required.
- Use the sleep timer (12.5) and motion trigger wake up (12.6) to automatically enter sleep mode during periods of inactivity.

## 13 Auxiliary port

This section is currently unavailable but can be updated on request.

## 14 Communication protocol

This section is currently unavailable. Users wishing to develop their own x-IMU communication interface are advised to study the open source x-IMU API (4.2). The API is source code is written in C# and well commented; porting the API (or aspects of) to another language should be a straight forward exercise.

## 15 Commands

Commands are executed by either sending a *command* packet to the x-IMU or by pressing the push button which may be configured to execute a specific command. Commands are sent using the x-IMU GUI via the *Commands* tab page (4.1.4). Once a command has been executed the x-IMU will echo the *command* packet back to the host computer as confirmation. As all communication from the x-IMU to the host computer is logged to the SD card (7), all command confirmations will be logged on the SD card. If a *command* packet is sent containing an invalid command code the x-IMU will respond with an *invalid command* error message (16.1.16).

Sending a *command* packet to the x-IMU will cause the x-IMU to momentarily pause sensor sampling and processing while the received data is processed. This may cause discrepancies in the otherwise fixed data output rates.

## 15.1 Individual commands

### 15.1.1 Null command

*Command code:* `0x0000`

*Description:* A *null command* is a valid command code but will result in no action. As all commands sent to the x-IMU are echoed back to the sender, a null command may be used by the host software to confirm communication with the x-IMU.

### 15.1.2 Reset

*Command code:* `0x0001`

*Description:* A *reset* command causes a software reset of the x-IMU. The x-IMU will close any open files on the SD card before reset, in this way the reset command may be of use to users wishing to break up a logging session into multiple files. The reset command is used put the x-IMU into bootloader (**??**) mode in order to upload new firmware. A *reset* command is sent by the x-IMU to the host computer as confirmation of reset, power on and wake up.

### 15.1.3 Sleep

*Command code:* `0x0002`

*Description:* A *sleep* command will put the device into sleep mode (12.4). The x-IMU will close any open files on the SD card before entering sleep mode. The x-IMU may be taken out of sleep mode by using the push button configured in *sleep/wake* mode or using the motion triggered wake up functionality (12.6).

### 15.1.4 Reset sleep timer

*Command code:* `0x0003`

*Description:* The *reset sleep timer* command will reset the sleep timer (12.5) countdown and so postpone sleep. An example usage of this command is to create behaviour where the x-IMU will automatically enter sleep mode when communication with the host software ends or connection is lost.

### 15.1.5 Sample gyroscope bias at temperature 1

*Command code:* `0x0004`

*Description:* The *sample gyroscope bias at temperature 1* command is used to calibrate the gyroscope bias parameters. This command should be sent while the x-IMU is stationary and at the lowest temperature the device is required to operate at. The x-IMU will measure the mean temperature and gyroscope output over approximately 16 seconds, store the results to registers (17.2.18 to 17.2.21) and then trigger a *calculate gyroscope bias parameters* command (15.1.7). The execution of this command will be aborted if a *gyroscope not stationary* error (16.1.17) occurs. See the gyroscope calibration section for more information (**??**).

### 15.1.6 Sample gyroscope bias at temperature 2

*Command code:* `0x0005`

*Description:* The *sample gyroscope bias at temperature 2* command is used to calibrate the gyroscope bias parameters. This command should be sent while the x-IMU is stationary and at the highest temperature the device is required to operate at. The x-IMU will measure the mean temperature and gyroscope output over approximately 16 seconds, store the results to registers (17.2.22 to 17.2.25) and then trigger a *calculate gyroscope bias parameters* command (15.1.7).The execution of this command will be aborted if a *gyroscope not stationary* error (16.1.17) occurs. See the gyroscope calibration section for more information (**??**).

### 15.1.7 Calculate gyroscope bias parameters

*Command code:* `0x0006`

*Description:* The *calculate gyroscope bias parameters* command is used to execute the on-board gyroscope bias calibration calculation. The algorithm uses the register values (17.2.18 to 17.2.25) previously sampled by the *sample gyroscope bias at temperature 1* (15.1.5) and *sample gyroscope bias at temperature 2* (15.1.6) commands to calculate the gyroscope bias parameters and update the registers (17.2.12 to 17.2.17). See the gyroscope calibration section for more information (**??**).

### 15.1.8 Look up accelerometer bias and sensitivity

*Command code:* `0x0007`

*Description:* The *look up accelerometer bias and sensitivity* command is used to set the accelerometer sensitivity (17.2.27 to 17.2.29) and bias registers (17.2.30 to 17.2.32) to the calibrated parameters defined by the accelerometer manufacturer. This command should be used each time the accelerometer full-scale range (17.2.26) is changed. See the accelerometer calibration section (**??**) for more information.

### 15.1.9 Measure magnetometer bias and sensitivity

*Command code:* `0x0008`

*Description:* The *measure magnetometer bias and sensitivity* command is used to run an on-board magnetometer calibration algorithm. The x-IMU use the magnetometer's internal field generator to measure the mean magnetometer bias and sensitivity over approximately 16 seconds independent of external magnetic interference. The magnetometer sensitivity (17.2.34 to 17.2.36) and bias (17.2.37 to 17.2.39) registers and then automatically updated. This command should be used each time the accelerometer full-scale range (17.2.33) is changed. The execution of this command will be aborted if a *magnetometer saturation* error (16.1.18) occurs. See the magnetometer calibration section (**??**) for more information.

### 15.1.10 Algorithm initialise

*Command code:* `0x0009`

*Description:* The *algorithm initialise* command will re-start the algorithm from initial conditions. This command can be used to 'force' the algorithm to converge to steady state conditions if previous distortions to magnetic or other extreme sensor measurements have left the IMU or AHRS algorithm output at an erroneous orientation. See the IMU/AHRS algorithms section (11) for more information.

### 15.1.11 Algorithm tare

*Command code:* `0x000A`

*Description:* The *algorithm tare* command is used to set the algorithm datum orientation and store the reference quaternion to the *tare quaternion* registers (17.2.50 to 17.2.53). These registers may be then be cleared using the clear tare command (15.1.12). See the IMU/AHRS algorithms section for more information (11).

### 15.1.12 Algorithm clear tare

*Command code:* `0x000B`

*Description:* The *algorithm clear tare* command is used to clear the *tare quaternion* registers (17.2.50 to 17.2.53) and return the datum orientation to alignment with the Earth coordinate frame.

#### 15.1.13 Algorithm initialise then tare

*Command code:* `0x000C`
*Description:* The *algorithm initialise then tare* command will execute an *algorithm initialise* command (15.1.10) and then execute an *algorithm clear tare* command (15.1.11) once the initialisation is complete.

# 16 Error messages

Error messages are sent by the x-IMU to warn the user or host software of any internal errors that have occurred. The x-IMU GUI will display these errors in message boxes for user acknowledgment. As all communication from the x-IMU to the host computer is logged to the SD card (7), all error messages will be logged on the SD card.

## 16.1 Individual error messages

#### 16.1.1 NoError

*Error code:* `0x0000`
*Description:* No error.

#### 16.1.2 Low battery

*Error code:* `0x0001`
*Description:* A *low battery* error is sent when the low battery voltage detection (12.3) detects that the battery voltage has fallen below the specific level defined in the *battery shutdown voltage* register (17.2.60). This message is sent immediately before the x-IMU enters sleep mode (12.4). See the low battery voltage detection section (12.3) for more information.

#### 16.1.3 USB receive buffer overrun

*Error code:* `0x0002`
*Description:* A *USB receive buffer over* error will be sent if the USB receive buffer overruns and data to be received was lost. This occurs when data is transmitted to the x-IMU at a rate greater than the rate it can be processed. Consider reducing the rate at which data is sent to the x-IMU if this error occurs repeatedly.

#### 16.1.4 USB transmit buffer overrun

*Error code:* `0x0003`
*Description:* A *transmit buffer overrun* error will be sent if the USB transmit buffer overruns and data due to be transmitted was lost. This will occur when the communication channel bandwidth is unable to cope with the amount of data being transmitted. Consider using lower data output rates if this error occurs repeatedly. This error may be ignored in applications where USB data is not essential and the SD card is the intended data output. In such applications, the user need only be concerned with *SD card write buffer overrun* errors (16.1.7). The x-IMU will attempt to transmit data via USB while the USB is detected as connected, if the USB is connect but the associated serial port not open then the USB transmit buffer will continue to overrun until the port is opened or the USB disconnected.

#### 16.1.5 Bluetooth receive buffer overrun

*Error code:* `0x0004`
*Description:* A *Bluetooth receive buffer over* error will be sent if the Bluetooth receive buffer overruns and data to be received was lost. This occurs when data is transmitted to the x-IMU at a rate greater than the rate it can be processed. Consider reducing the rate at which data is sent to the x-IMU if this error occurs repeatedly.

#### 16.1.6 Bluetooth transmit buffer overrun

*Error code:* `0x0005`

*Description:* A *transmit buffer overrun* error will be sent if the Bluetooth transmit buffer overruns and data due to be transmitted was lost. This will occur when the communication channel bandwidth is unable to cope with the amount of data being transmitted. Consider using lower data output rates if this error occurs repeatedly. *Transmit buffer overrun* errors may be expected in the Bluetooth communication channel quality deteriorates; for example, if out of range. This error may be ignored in applications where the Bluetooth data is not essential and the SD card is the intended data output. In such applications, the user need only be concerned with *SD card write buffer overrun* errors (16.1.7).

#### 16.1.7 SD card write buffer overrun

*Error code:* `0x0006`

*Description:* An *SD card buffer over* error will be sent if the SD card buffer is overrun and data to be written to the SD was lost. Consider using lower data output rates if this error occurs repeatedly. An occurrence of this error may go unnoticed while the USB and Bluetooth are not used. The red SD card LED (3.3.4) indicates SD card activity, if this LED behaviour approaches that of being solidly on then it is likely that the this error is occurring.

#### 16.1.8 Too few bytes in packet

*Error code:* `0x0007`

*Description:* A *too few bytes in packet* error will be sent if the received packet does not contain enough bytes to be valid. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.9 Too many bytes in packet

*Error code:* `0x0008`

*Description:* A *too many bytes in packet* error will be sent if the received packet does not contains too many bytes to be valid. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.10 Invalid checksum

*Error code:* `0x0009`

*Description:* An *invalid checksum* error will be sent if the received packet contains a valid number of bytes but contains an invalid checksum. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.11 Unknown packet header

*Error code:* `0x000A`

*Description:* An *Unknown packet header* error will be sent if the received packet contains a valid number of bytes and checksum but the header is not recognised. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.12 Invalid number of bytes for packet header

*Error code:* `0x000B`

*Description:* An *invalid number of bytes for packet header* error will be sent if the received packet contains a valid number of bytes, checksum and packet header but the number of bytes does not match that expected for the specific packet header. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.13 Invalid register address

*Error code:* `0x000C`
*Description:* An *invalid register address* error will be sent if the read or write register packet contains an invalid register address. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.14 Register read-only

*Error code:* `0x000D`
*Description:* A *register read-only* error will be sent if the write register packet represents an attempt to write a read-only register. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.15 Invalid register value

*Error code:* `0x000E`
*Description:* An *invalid register value* error will be sent if the write register packet contains an invalid register value for the specific address. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.16 Invalid command

*Error code:* `0x000F`
*Description:* An *invalid command* error will be sent if the command code within the command packet is not valid. This error is only relevant users developing their own communication software and not using the x-IMU API or x-IMU GUI.

#### 16.1.17 Gyroscope not stationary

*Error code:* `0x0010`
*Description:* A *gyroscope not stationary* error will be sent if the gyroscope was detected as not being stationary during the execution of a *sample gyroscope bias* command (15.1.5 or 15.1.6) and the execution of the process was aborted. See the gyroscope calibration section (**??**) for more information.

#### 16.1.18 Magnetometer saturation

*Error code:* `0x0011`
*Description:* A *magnetometer saturation* error will be sent if the measurements taken during the execution of the *measure magnetometer bias and sensitivity* command (15.1.9) were detected as having saturated and the execution of the command was aborted. See the magnetometer calibration section (**??**) for more information.

#### 16.1.19 Incorrect auxiliary port mode

*Error code:* `0x0012`
*Description:* An *incorrect auxiliary port mode* error will be sent if an auxiliary port action is requested while the auxiliary port is not in the correct mode (17.2.64) for that action. For example, an *incorrect auxiliary port mode* error will be sent if a *digital IO data* packet is received while the auxiliary port mode is *Disabled*. See the auxiliary port section (13) for more information.

## 17 Registers

All x-IMU settings are saved in a bank of registers stored to non-volatile flash memory. Each register has a 16-bit address and 16-bit value. These values may be viewed, edited, read, written and backed up to file

using the x-IMU GUI via the *registers* tab page.

## 17.1 Reading and writing registers

Any register may be read by sending a *read register* packet containing register address to be read. The x-IMU will respond with a *register write* packet containing the register address and value. The x-IMU will automatically send all register values so that settings are hardcoded as the first packets written to the SD card (7). Registers may be written by sending a *register write* packet containing the register address to be written to and the new register value. The x-IMU will respond with a *register write* packet containing the register address and new value.

If a *read register* or *register write* packet is sent containing an invalid register address the x-IMU will respond with an *invalid register address* error message (16.1.13). If a *register write* packet is sent containing an read-only register address the x-IMU will respond with a *register read-only* error message (16.1.14). If a *register write* packet is sent containing register value that is invalid for the specified address the x-IMU will respond with an invalid register value error message (16.1.15).

Sending a *read register* or *register write* packet to the x-IMU will cause the x-IMU to momentarily pause sensor sampling and processing while the packet is processed. This may cause discrepancies in the otherwise fixed data output rates.

## 17.2 Individual register details

### 17.2.1 Firmware version major number

*Address:* `0x0000`
*Value:* 0 to 65535. Read-only.
*Description:* The major number of the current firmware version loaded on the x-IMU.

### 17.2.2 Firmware version minor number

*Address:* `0x0001`
*Value:* 0 to 65535. Read-only.
*Description:* The minor number of the current firmware version loaded on the x-IMU.

### 17.2.3 Device ID

*Address:* `0x0002`
*Value:* `0x0000` to `0xFFFF`. Read-only.
*Description:* The unique 4 hexadecimal digit ID of the x-IMU taken as the last 2 bytes of the Bluetooth MAC address.

### 17.2.4 Button mode

*Address:* `0x0003`
*Value:*
`0x0000` = Disabled
`0x0001` = *Reset* command (15.1.2)
`0x0002` = *Sleep/Wake* command (15.1.3)
`0x0003` = *Algorithm initialise* command (15.1.10)
`0x0004` = *Algorithm tare* command (15.1.11)
`0x0005` = *Algorithm initialise then tare* command (15.1.13)
*Description:* The command to be executed when the push button (3.2) is pressed. See the commands section for more information on individual commands.

#### 17.2.5 Battery voltmeter sensitivity

*Address:* `0x0004`
*Value:* Q11.5 signed fixed point value between −1024 and +1024.
*Description:* Calibrated sensitivity of the battery ADC in V/lsb. See parameter $s_v$ in the battery voltmeter section (9.1). The typical value is 621 V/lsb.

#### 17.2.6 Battery voltmeter bias

*Address:* `0x0005`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the battery ADC in lsb. See parameter $b_v$ in the battery voltmeter section (9.1). The typical value is 0 lsb.

#### 17.2.7 Thermometer sensitivity

*Address:* `0x0006`
*Value:* Q10.6 signed fixed point value between −512 and +512.
*Description:* Calibrated sensitivity of the thermometer in °C/lsb. See parameter $s_\tau$ in the thermometer section (9.2). The typical value is provided as 280 °C/lsb in the IMU-3000[3] datasheet.

#### 17.2.8 Thermometer bias

*Address:* `0x0007`
*Value:* Q16.0 signed fixed point value between −32768 and +32677.
*Description:* Calibrated bias of the thermometer in lsb. See parameter $b_\tau$ (9.2) in the thermometer section (9.2). The typical value is provided as $-23,000$ lsb in the IMU-3000[3] datasheet.

#### 17.2.9 Gyroscope x-axis sensitivity

*Address:* `0x0008`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated sensitivity of the gyroscope x-axis in °/s/lsb. See parameter $s_{g_x}$ in the gyroscope section (9.3). The typical value is provided as 16.4 °/s/lsb. in the IMU-3000[3] datasheet.

#### 17.2.10 Gyroscope y-axis sensitivity

*Address:* `0x0009`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated sensitivity of the gyroscope y-axis in °/s/lsb. See parameter $s_{g_y}$ in the gyroscope section (9.3). The typical value is provided as 16.4 °/s/lsb. in the IMU-3000[3] datasheet.

#### 17.2.11 Gyroscope z-axis sensitivity

*Address:* `0x000A`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated sensitivity of the gyroscope z-axis in °/s/lsb. See parameter $s_{g_z}$ in the gyroscope section (9.3). The typical value is provided as 16.4 °/s/lsb. in the IMU-3000[3] datasheet.

#### 17.2.12 Gyroscope x-axis bias

*Address:* `0x000B`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the gyroscope x-axis at 25 °C in lsb. See parameter $b_{g_x}$ in the gyroscope section (9.3). The typical value is provided as 0 lsb in the IMU-3000[3] datasheet. The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.13 Gyroscope y-axis bias

*Address:* `0x000C`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the gyroscope y-axis at 25 °C in lsb. See parameter $b_{g_y}$ in the gyroscope section (9.3). The typical value is provided as 0 lsb in the IMU-3000[4] datasheet. The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.14 Gyroscope z-axis bias

*Address:* `0x000D`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the gyroscope z-axis at 25 °C in lsb. See parameter $b_{g_z}$ in the gyroscope section (9.3). The typical value is provided as 0 lsb in the IMU-3000[4] datasheet. The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.15 Gyroscope x-axis bias temperature sensitivity

*Address:* `0x000E`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated bias temperature sensitivity of the gyroscope x-axis in lsb/°C. See parameter $f_x$ in the gyroscope section (9.3). The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.16 Gyroscope y-axis bias temperature sensitivity

*Address:* `0x000F`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated bias temperature sensitivity of the gyroscope y-axis in lsb/°C. See parameter $f_y$ in the gyroscope section (9.3). The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.17 Gyroscope z-axis bias temperature sensitivity

*Address:* `0x0010`
*Value:* Q6.10 signed fixed point value between −32 and +32.
*Description:* Calibrated bias temperature sensitivity of the gyroscope z-axis in lsb/°C. See parameter $f_z$ in the gyroscope section (9.3). The x-IMU has build in algorithms to accuracy evaluate this parameter automatically using the *Calculate gyroscope bias parameters* command (15.1.7), see the gyroscope calibration section for more information (**??**).

#### 17.2.18 Gyroscope sample 1 - Temperature

*Address:* `0x0011`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Temperature at gyroscope bias sample 1 in °C. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 1 is recorded using the *Sample gyroscope bias at temperature 1* command (15.1.5). See the gyroscope calibration section for more information (**??**).

#### 17.2.19 Gyroscope sample 1 - x-axis bias

*Address:* `0x0012`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* x-axis bias at gyroscope bias sample 1 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 1 is recorded using the *Sample gyroscope bias at temperature 1* command (15.1.5). See the gyroscope calibration section for more information (**??**).

#### 17.2.20 Gyroscope sample 1 - y-axis bias

*Address:* `0x0013`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* y-axis bias at gyroscope bias sample 1 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 1 is recorded using the *Sample gyroscope bias at temperature 1* command (15.1.5). See the gyroscope calibration section for more information (**??**).

#### 17.2.21 Gyroscope sample 1 - z-axis bias

*Address:* `0x0014`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* z-axis bias at gyroscope bias sample 1 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 1 is recorded using the *Sample gyroscope bias at temperature 1* command (15.1.5). See the gyroscope calibration section for more information (**??**).

#### 17.2.22 Gyroscope sample 2 - Temperature

*Address:* `0x0015`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Temperature at gyroscope bias sample 2 in °C. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 2 is recorded using the *Sample gyroscope bias at temperature 2* command (15.1.6). See the gyroscope calibration section for more information (**??**).

#### 17.2.23 Gyroscope sample 2 - x-axis bias

*Address:* `0x0016`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* x-axis bias at gyroscope bias sample 2 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 2 is recorded using the *Sample gyroscope bias at temperature 2* command (15.1.6). See the gyroscope calibration section for more information (**??**).

#### 17.2.24 Gyroscope sample 2 - y-axis bias

*Address:* `0x0017`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* y-axis bias at gyroscope bias sample 2 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 2 is recorded using the *Sample gyroscope bias at temperature 2* command (15.1.6). See the gyroscope calibration section for more information (**??**).

#### 17.2.25 Gyroscope sample 2 - z-axis bias

*Address:* `0x0018`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* z-axis bias at gyroscope bias sample 2 in lsb. Gyroscope bias samples 1 and 2 are required for the automatic calculation of the gyroscope bias parameters (17.2.12 to 17.2.17) using the *Calculate gyroscope bias parameters* command (15.1.7). The gyroscope bias sample 2 is recorded using the *Sample gyroscope bias at temperature 2* command (15.1.6). See the gyroscope calibration section for more information (**??**).

#### 17.2.26 Accelerometer full-scale

*Address:* `0x0019`
*Value:* `0x0000` = ±2 g
`0x0001` = ±4 g
`0x0002` = ±8 g
*Description:* Sets the full-scale range of the accelerometer. Each full-scale range will have different associated sensitivity (17.2.27 to 17.2.29) and bias (17.2.30 to 17.2.32) values. The *Look up accelerometer bias and sensitivity* command (15.1.8) may be used to recall the correct calibrated sensitivity and bias values for the selected full-scale range.

#### 17.2.27 Accelerometer x-axis sensitivity

*Address:* `0x001A`
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the accelerometer x-axis in g/lsb. See parameter $s_{a_x}$ in the accelerometer section (9.4). The typical values are provided for each full-scale range (17.2.26) in the LSM303DLH datasheet. These values can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.28 Accelerometer y-axis sensitivity

*Address:* `0x001B`
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the accelerometer y-axis in g/lsb. See parameter $s_{a_y}$ in the accelerometer section (9.4). The typical values are provided for each full-scale range (17.2.26) in the LSM303DLH datasheet. These values can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.29 Accelerometer z-axis sensitivity

*Address:* `0x001C`
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the accelerometer z-axis in g/lsb. See parameter $s_{a_z}$ in the accelerometer section (9.4). The typical values are provided for each full-scale range (17.2.26) in the LSM303DLH datasheet. These values can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.30 Accelerometer x-axis bias

*Address:* `0x001D`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the accelerometer x-axis in lsb. See parameter $b_{a_x}$ in the accelerometer section (9.4). A typical value of 0 lsb is provided in the LSM303DLH datasheet. This value can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.31 Accelerometer y-axis bias

*Address:* `0x001E`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the accelerometer y-axis in lsb. See parameter $b_{a_y}$ in the accelerometer section (9.4). A typical value of 0 lsb is provided in the LSM303DLH datasheet. This value can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.32 Accelerometer z-axis bias

*Address:* `0x001F`
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the accelerometer z-axis in lsb. See parameter $b_{a_z}$ in the accelerometer section (9.4). A typical value of 0 lsb is provided in the LSM303DLH datasheet. This value can be set automatically using the *Look up accelerometer bias and sensitivity* command (15.1.8).

#### 17.2.33 Magnetometer full-scale

*Address:* `0x0020`
*Value:* `0x0000` = ±1.3 G
`0x0001` = ±1.9 G
`0x0002` = ±2.5 G
`0x0003` = ±4.0 G
`0x0004` = ±4.7 G
`0x0005` = ±5.6 G
`0x0006` = ±8.1 G
*Description:* Sets the full-scale range of the magnetometer. Each full-scale range will have different associated sensitivity (17.2.34 to 17.2.36) and bias (17.2.37 to 17.2.39) values. The *Measure magnetometer bias and sensitivity* command (15.1.9) may be used to measure the correct calibrated sensitivity and bias values for the selected full-scale range using an on-board algorithm.

#### 17.2.34 Magnetometer x-axis sensitivity

*Address:* `0x0021`
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the magnetometer x-axis in G/lsb. See parameter $s_{m_x}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.35 Magnetometer y-axis sensitivity

*Address:* 0x0022
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the magnetometer y-axis in G/lsb. See parameter $s_{m_y}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.36 Magnetometer z-axis sensitivity

*Address:* 0x0023
*Value:* Q11.4 signed fixed point value between −2048 and +2048.
*Description:* Calibrated sensitivity of the magnetometer z-axis in G/lsb. See parameter $s_{m_z}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.37 Magnetometer x-axis bias

*Address:* 0x0024
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the magnetometer x-axis in lsb. See parameter $b_{m_x}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.38 Magnetometer y-axis bias

*Address:* 0x0025
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the magnetometer y-axis in lsb. See parameter $b_{m_y}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.39 Magnetometer z-axis bias

*Address:* 0x0026
*Value:* Q8.8 signed fixed point value between −128 and +128.
*Description:* Calibrated bias of the magnetometer z-axis in lsb. See parameter $b_{m_z}$ in the magnetometer section (9.5). The typical values are provided for each full-scale range (17.2.33) in the LSM303DLH datasheet. Accurate values can be set automatically using the *Measure magnetometer bias and sensitivity* command (15.1.9).

#### 17.2.40 Magnetometer x-axis hard-iron bias

*Address:* 0x0027
*Value:* Q5.11 signed fixed point value between −16 and +16.
*Description:* Calibrated hard-iron bias affecting the magnetometer x-axis in G. See parameter $h_x$ in the magnetometer section (9.5). The hard-iron bias parameters will change when the x-IMU's local magnetic environment is altered; for example, when the x-IMU is fixed to the battery. Use of the magnetometer and AHRS algorithm requires that the hard-iron bias parameters are accurately calibrated. See the hard-iron calibrations section for more information (**??**).

#### 17.2.41 Magnetometer y-axis hard-iron bias

*Address:* `0x0028`
*Value:* Q5.11 signed fixed point value between −16 and +16.
*Description:* Calibrated hard-iron bias affecting the magnetometer y-axis in G. See parameter $h_x$ in the magnetometer section (9.5). The hard-iron bias parameters will change when the x-IMU's local magnetic environment is altered; for example, when the x-IMU is fixed to the battery. Use of the magnetometer and AHRS algorithm requires that the hard-iron bias parameters are accurately calibrated. See the hard-iron calibrations section for more information (**??**).

#### 17.2.42 Magnetometer z-axis hard-iron bias

*Address:* `0x0029`
*Value:* Q5.11 signed fixed point value between −16 and +16.
*Description:* Calibrated hard-iron bias affecting the magnetometer z-axis in G. See parameter $h_x$ in the magnetometer section (9.5). The hard-iron bias parameters will change when the x-IMU's local magnetic environment is altered; for example, when the x-IMU is fixed to the battery. Use of the magnetometer and AHRS algorithm requires that the hard-iron bias parameters are accurately calibrated. See the hard-iron calibrations section for more information (**??**).

#### 17.2.43 Algorithm mode

*Address:* `0x002A`
*Value:* `0x0000` = Disabled
`0x0000` = IMU
`0x0001` = AHRS
*Description:* Sets the algorithm mode, see the IMU/AHRS algorithms section for more information (11). The algorithm mode can be set to *Disabled* to reduce power consumption when the algorithms are not required.

#### 17.2.44 Algorithm proportional gain

*Address:* `0x002B`
*Value:* Q8.8 signed fixed point value between 0 and +128.
*Description:* Algorithm proportional feedback gain. The proportional gain governs the rate at which the algorithm output converges to an orientation assumed by the accelerometer and magnetometer; a low value will 'trust' the gyroscope data more and the accelerometer and magnetometer less, a high value will 'trust' the gyroscope less and the accelerometer and magnetometer more. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.45 Algorithm integral gain

*Address:* `0x002C`
*Value:* Q2.14 signed fixed point value between 0 and +2.
*Description:* Algorithm integral feedback gain. The integral gain governs the rate at which the algorithm compensates for gyroscope bias drift. It is recommended that users ensure accurate gyroscope bias temperature sensitivity calibration and use an integral feedback gain of 0 to avoid instabilities. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.46 Algorithm initial proportional gain

*Address:* `0x002D`
*Value:* Q10.6 signed fixed point value between 0 and 512.
*Description:* Algorithm initial proportional feedback gain used each time the algorithm resets. The effective proportional gain will ramp down from the initial proportional gain to the proportional gain 17.2.44 over a specified period of time 17.2.47. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.47 Algorithm initilasation period

*Address:* `0x002E`
*Value:* Q8.8 signed fixed point value between 0 and +128.
*Description:* Algorithm ramp down period (in seconds) used each time the algorithm resets. The effective proportional gain will ramp down from the initial proportional gain (17.2.44) to the proportional gain 17.2.44 over a specified period of time 17.2.47. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.48 Algorithm minimum valid magnetic field magnitude

*Address:* `0x0041`
*Value:* Q5.11 signed fixed point value between −16 and +16.
*Description:* The minimum valid magnetic field magnitude (in G) that may be used by the algorithm in the estimation of heading. Magnetic fields of an invalid magnitude will be ignored by the AHRS algorithm so that heading is determined from gyroscope measurements alone. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.49 Algorithm maximum valid magnetic field magnitude

*Address:* `0x0042`
*Value:* Q5.11 signed fixed point value between −16 and +16.
*Description:* The maximum valid magnetic field magnitude (in G) that may be used by the algorithm in the estimation of heading. Magnetic fields of an invalid magnitude will be ignored by the AHRS algorithm so that heading is determined from gyroscope measurements alone. See the IMU/AHRS algorithms section for more information (11).

#### 17.2.50 Tare quaternion (element 0)

*Address:* `0x002F`
*Value:* Q1.15 signed fixed point value between −1 and +1.
*Description:* Quaternion stored to compute the algorithm output after a tare operation **??** has been preformed. The tare quaternion can be set using the *tare* command (**??**) and cleared using the *clear tare* command (**??**). See the IMU/AHRS algorithms section for more information (11).

#### 17.2.51 Tare quaternion (element 1)

*Address:* `0x0030`
*Value:* Q1.15 signed fixed point value between −1 and +1.
*Description:* Quaternion stored to compute the algorithm output after a tare operation **??** has been preformed. The tare quaternion can be set using the *tare* command (**??**) and cleared using the *clear tare* command (**??**). See the IMU/AHRS algorithms section for more information (11).

#### 17.2.52 Tare quaternion (element 2)

*Address:* `0x0031`
*Value:* Q1.15 signed fixed point value between −1 and +1.
*Description:* Quaternion stored to compute the algorithm output after a tare operation **??** has been preformed. The tare quaternion can be set using the *tare* command (**??**) and cleared using the *clear tare* command (**??**). See the IMU/AHRS algorithms section for more information (11).

#### 17.2.53 Tare quaternion (element 3)

*Address:* 0x0032
*Value:* Q1.15 signed fixed point value between −1 and +1.
*Description:* Quaternion stored to compute the algorithm output after a tare operation **??** has been preformed. The tare quaternion can be set using the *tare* command (**??**) and cleared using the *clear tare* command (**??**). See the IMU/AHRS algorithms section for more information (11).

#### 17.2.54 Sensor data mode

*Address:* 0x0033
*Value:* 0x0000 = Raw ADC results
0x0001 = Calibrated measurements
*Description:* Sets the data output mode of the sensors. See the sensors section for more details (9).

#### 17.2.55 Date/time data output rate

*Address:* 0x0034
*Value:* 0x0000 = Disabled (sent on reset/wake only)
0x0001 = 1 Hz
*Description:* Output rate of the *date/time data* packets. Data rates can be reduced or disabled to conserve power.

#### 17.2.56 Battery and thermometer data output rate

*Address:* 0x0035
*Value:* 0x0000 = Disabled
0x0001 = 1 Hz
0x0002 = 2 Hz
0x0003 = 4 Hz
0x0004 = 8 Hz
0x0005 = 16 Hz
0x0006 = 32 Hz
0x0007 = 64 Hz
0x0008 = 128 Hz
0x0009 = 256 Hz
*Description:* Output rate of the *battery and thermometer data* packets. Data rates can be reduced or disabled to conserve power.

#### 17.2.57 Inertial and magnetic data output rate

*Address:* 0x0036
*Value:* 0x0000 = Disabled
0x0001 = 1 Hz
0x0002 = 2 Hz
0x0003 = 4 Hz
0x0004 = 8 Hz
0x0005 = 16 Hz
0x0006 = 32 Hz
0x0007 = 64 Hz
0x0008 = 128 Hz
0x0009 = 256 Hz
*Description:* Output rate of the *inertial and magnetic data* packets. Data rates can be reduced or disabled to conserve power.

#### 17.2.58 Quaternion data output rate

*Address:* 0x0037
*Value:* 0x0000 = Disabled
0x0001 = 1 Hz
0x0002 = 2 Hz
0x0003 = 4 Hz
0x0004 = 8 Hz
0x0005 = 16 Hz
0x0006 = 32 Hz
0x0007 = 64 Hz
0x0008 = 128 Hz
0x0009 = 256 Hz
*Description:* Output rate of the *quaternion data* packets. Data rates can be reduced or disabled to conserve power.

#### 17.2.59 SD card new file name

*Address:* 0x0038
*Value:* 00000 to 65535
*Description:* The file name used to be used when the next file is created on the SD card. See the SD card section for more information (7).

#### 17.2.60 Battery shotdown voltage

*Address:* 0x0039
*Value:* Q4.12 signed fixed point value between 3.5 and +8.
*Description:* Minimum voltage threshold for the device to shutdown. See the low battery voltage detection section for more information (12.3).

#### 17.2.61 Sleep timer

*Address:* 0x003A
*Value:* 0 to 65535
*Description:* Sleep timer value in seconds. Once this period has elapsed, the x-IMU will enter sleep mode (12.4). A value of 0 seconds will disable the sleep timer. See the sleep timer (12.5) section for more information.

#### 17.2.62 Motion trigger wake up

*Address:* 0x003B
*Value:* 0x0000 = Disabled
0x0001 = Low sensitivity
0x0001 = High sensitivity
*Description:* Enables the sensitivity of the motion trigger wake up. See the motion trigger wake up section for more information (12.6).

#### 17.2.63 Bluetooth power

*Address:* 0x003C
*Value:* 0x0000 = Disabled
0x0001 = Enabled
*Description:* Enables or disables Bluetooth. The Bluetooth can be disabled to conserve power.

#### 17.2.64 Auxiliary port mode

*Address:* `0x003D`
*Value:* `0x0000` = Disabled
`0x0001` = Digital I/O
*Description:* Sets the auxiliary port mode. See the auxiliary port section for more information (13).

#### 17.2.65 Digital I/O direction

*Address:* `0x003F`
*Value:* `0x0000` = All channels are inputs
`0x0001` = Channels 0, 1, 2, 3, 4, 5 and 6 are inputs, 7 is an output
`0x0002` = Channels 0, 1, 2, 3, 4 and 5 are inputs, 6 and 7 are outputs
`0x0003` = Channels 0, 1, 2, 3 and 4 are inputs, 5, 6 and 7 are outputs
`0x0004` = Channels 0, 1, 2 and 3 are inputs, 4, 5, 6 and 7 are outputs
`0x0005` = Channels 0, 1 and 2 are inputs, 3, 4, 5, 6 and 7 are outputs
`0x0006` = Channels 0 and 1 are inputs, 2, 3, 4, 5, 6 and 7 are outputs
`0x0007` = Channel 0 is an input, 1, 2, 3, 4, 5, 6 and 7 are outputs
`0x0008` = All channels are outputs
*Description:* Sets the direction of the auxiliary port channels when in digital I/O mode. See the digital I/O mode section for more information (**??**).

#### 17.2.66 Digital I/O data output rate

*Address:* `0x0040`
*Value:* `0x0000` = On change only
`0x0001` = 1 Hz
`0x0002` = 2 Hz
`0x0003` = 4 Hz
`0x0004` = 8 Hz
`0x0005` = 16 Hz
`0x0006` = 32 Hz
`0x0007` = 64 Hz
`0x0008` = 128 Hz
`0x0009` = 256 Hz
*Description:* Output rate of the digital I/O data packets. See the digital I/O mode section for more information (**??**).